Evaluation and Further Development of a Fire Control and Information GUI

Karin Fossum

December 10, 2005
Master's Thesis in Computing Science, 20 credits
Supervisor at CS-UmU: Lena Palmquist
Examiner: Per Lindström

Umeå University
Department of Computing Science
SE-901 87 UMEÅ
SWEDEN

Abstract

This master thesis embrace three evaluation methods: SGT, heuristic evaluation, and user test. The evaluations were made on a fire control and information application that in the future will be used in BAE Land Systems Hägglunds' combat vehicles. In the SGT and heuristic evaluation an earlier prototype was tested and then a new design proposition of the interface was made. The design proposition was then realized in an interactive software-based prototype. This prototype was tested by users in a user test which revealed more design flaws. From the result of the user test a second design proposition was made and also realized in a prototype. The conclusion is that it is recommended to use several evaluation methods to discover different kinds of problems in the interface. Another conclusion is that it is important to involve the users in the design process.

Key Words

GUI, interaction design, HCI, usability, evaluation.

Utvärdering och Vidareutveckling av en Eldlednings- och Informationsapplikation

Sammanfattning

Detta examensarbete behandlar tre utvärderingsmetoder: SGT, heuristisk utvärdering och användartest. Utvärderingarna gjordes på en eldlednings- och informationsapplikation som i framtiden kommer att användas i BAE Land Systems Hägglunds stridsfordon. I SGT:n och den heuristiska utvärderingen testades en tidigare prototyp och ett nytt designförslag för gränssnittet togs fram. Designförslaget blev sedan realiserat i en interaktiv mjukvaru-prototyp. Denna prototyp testades av användarna i ett användartest vilket avslöjade fler designmissar. Ur resultatet av användartestet skapades ett andra designförslag vilket också blev realiserat i en prototyp. Slutsatsen är att det rekommenderas att använda flera utvärderingsmetoder för att hitta olika typer av problem i gränssnittet. En annan slutsats är att det är viktigt att inkludera användarna i designprocessen.

Nyckelord

GUI, interaktionsdesign, MDI (människa-dator interaktion), användbarhet, utvärdering.

Preface

During my years of study my interest in human-computer interaction has accelerated. The combination of working for the users to make their day easier and at the same time being on the front edge of technology brings me a lot of inspiration to do good work. The opportunity to meet and talk to people and use the computer as a tool to realize ideas is stimulating.

Knowing the importance of user influence in the design work was something that I have brought with me from my four years of study at the Master of Science program Interaction and Design at Umeå University, Sweden. The result of this study strengthens my opinion of the importance of the user's involvement in the design process.

The assigner, BAE Land Systems Hägglunds, is one of the world leading companies in the construction of combat vehicles. Military systems often lead the technology development and therefore this branch is attractive for those with an interest in new technology - which I have. Furthermore Hägglunds has also understood the importance of human-computer interaction. Therefore I am really thankful to have had the opportunity to do this project.

This report is part of a master thesis project and one requirement for a degree of Master of Science in Interaction Technology and Design.

Örnsköldsvik, November 2005

Contents

1 Introduction **1**
- 1.1 The Assigner . . . 1
- 1.2 Task Description . . . 2
- 1.3 Target Group . . . 2
- 1.4 The Application . . . 2

2 Task, Resources and Goal **3**
- 2.1 Task . . . 3
- 2.2 Resources . . . 3
  - 2.2.1 Preconditions . . . 3
- 2.3 Goal . . . 4
- 2.4 Work Procedure . . . 4
- 2.5 Output . . . 4

3 Methods and Materials **5**
- 3.1 Evaluation Paradigms . . . 5
  - 3.1.1 Predictive Evaluation . . . 6
  - 3.1.2 Usability Testing . . . 7
- 3.2 Evaluation Techniques . . . 7
  - 3.2.1 Model User's Task Performance . . . 7
  - 3.2.2 Ask Experts . . . 8
  - 3.2.3 Ask Users . . . 8
- 3.3 Evaluation Methods . . . 8
  - 3.3.1 Task Analysis . . . 9
  - 3.3.2 Heuristic Evaluation . . . 13
  - 3.3.3 User Tests . . . 19
- 3.4 Prototyping . . . 20
- 3.5 The DECIDE Framework . . . 22
  - 3.5.1 Determine the Overall Goals . . . 22
  - 3.5.2 Explore the Questions . . . 23
  - 3.5.3 Choose the Evaluation Paradigm and Techniques . . . 23

3.5.4 Identify the Practical Issues . . . 23
3.5.5 Decide How to Deal With the Ethical Issues . . . 25
3.5.6 Evaluate, Interpret, and Present the Data . . . 25

4 Results **27**
4.1 SGT . . . 27
4.2 Heuristic Evaluation . . . 27
4.2.1 Design Principles . . . 28
4.2.2 Usability Principles . . . 28
4.2.3 A Template for Heuristic Evaluation . . . 29
4.3 The First Design Proposition . . . 29
4.3.1 Changes . . . 29
4.3.2 Additions . . . 30
4.3.3 Removals . . . 30
4.4 User Test . . . 31
4.4.1 The Questionnaire . . . 31
4.4.2 The Interview . . . 31
4.5 The Second Design Proposition . . . 31
4.5.1 Changes . . . 32
4.5.2 Additions . . . 32
4.5.3 Removals . . . 32

5 Conclusions **33**
5.1 Limitations . . . 34
5.1.1 SGT . . . 34
5.1.2 Heuristic Evaluation . . . 34
5.1.3 User Test . . . 34
5.2 Future Work . . . 34

6 Acknowledgements **35**

References **37**

A A Template for Heuristic Evaluation **39**
A.1 Graphics . . . 39
A.1.1 How to Use Color . . . 39
A.1.2 Simplicity . . . 39
A.1.3 Consistency . . . 40
A.1.4 Constraints . . . 40
A.2 Structure . . . 40
A.2.1 Simplicity . . . 40
A.2.2 Avoid Dead-Ends . . . 40

A.2.3 Avoid Narrow Menus . . . 41
A.2.4 Consistency . . . 41
A.3 Cognitive Help for the User . . . 41
A.3.1 Safety . . . 41
A.3.2 Learnability/Effectiveness . . . 41
A.3.3 Memorability/Efficiency . . . 41
A.3.4 Error Messages . . . 42
A.3.5 Visibility and Feedback . . . 42
A.3.6 Mapping . . . 42
A.3.7 Language . . . 42
A.4 Navigation Support . . . 43
A.4.1 Guidance . . . 43
A.4.2 Exploration . . . 43
A.4.3 Flexibility . . . 43
A.4.4 Error Prevention . . . 43
A.4.5 Constraints . . . 44
A.5 System Knowledge . . . 44
A.5.1 Utility . . . 44

List of Figures

3.1 Graphical representation of HTA for borrowing a book from the library [23, p. 233]. 11
3.2 Text-based HTA for borrowing a book from the library [23, p. 233]. . . 11
3.3 Graphical representation of SGT for finding an article on the internet. . 12
3.4 Text-based SGT for finding an article on the internet. 12
3.5 Example of a logical constraint. The Undo action is inactivated because it is not possible/allowed to perform the Undo action. 14
3.6 Example of mapping. Four different placements of controls. Based on Figure 1.10 in [23, p. 23]. 14
3.7 The design-prototype-test cycle [26]. 21
3.8 Illustration of the relation between the designer's model, the user's mental model and the image the system presents. Based on Figure 1.10 in [19, p. 16]. 23
3.9 The different types of buttons in the prototype. 24

4.1 Illustration of the save symbol. 32

A.1 Example of mapping. Four different placements of controls. 43
A.2 Example of a logical constraint. The Undo action is inactivated because it is not possible/allowed to perform the Undo-action. 44

List of Tables

3.1 A summary of the paradigms, techniques, and methods used. 5
3.2 Characteristics of different evaluation paradigms. Based on Table 11.1 in [23, p. 344]. 6
3.3 Subgoal templates (SGTs), specific task elements, and information requirements. Based on Table 17.2 in [21, p. 356]. 10
3.4 Sequencing elements used in SGT. Based on Table 13.3 in [21, p. 359]. . 11
3.5 A summarize of Nielsen's survey [14]. 18

Chapter 1

Introduction

The word "interaction" has been more widely used the past years. Interactive products are all over the market today; for example small electronic balls which, by asking questions, can figure out what you are thinking about and the fashionable product "interactive television". All these products are made to serve the user's expectations and demands. Therefore the area of interaction design is closely related to usability. Developing a GUI (graphical user interface) must therefore focus on the user's interests. However, since interaction design experts are specialized in usability and have knowledge and experience in the area, they can serve as evaluators when a GUI is developed. There are many benefits in using experts when evaluating a system, but the end user's capability to discover problems should never be underestimated.

BAE Land Systems Hägglunds is the assigner of this master thesis and will from now on be referred to as Hägglunds. Hägglunds have discovered the importance of making their products usable from the user's point of view. This project is focused on making usability improvements in a system provided by Hägglunds. The improvements, in the form of a new design proposistion, were discovered in three evaluations, where one included the users. To be able to test the system on the users, an early design suggestion was implemented in an interactive software-based prototype. The prototype was a useful communication device when discussing the GUI improvements with the users and other stakeholders.

1.1 The Assigner

An important part of the development of combat vehicles for Hägglunds is developing the interface between man and machine. To make the developing process more efficient, a number of prototypes are constructed. These serve both as communication devices towards the customer and as a design and construction base. To be able to guarantee that the vehicle's MMI (man-machine interface) is serving its function, it is important that the evaluations made, present the best possible view of the system's advantages and disadvantages so an efficient further development is possible.

1.2 Task Description

The task is to develop a design proposition to a fire control and information application. An earlier prototype, WCS, exists which serves as input to this project. Three evaluation methods were used to recognize design flaws, namely SGT, heuristic evaluation, and user test. A prototype was developed and tested with end users and people with knowledge of system. Based on the user test a new design was proposed and also implemented in a software-based prototype.

1.3 Target Group

The purchaser of this system is the Swedish army. The specific users of the end system will get a thorough course in handling the system. Therefore only people educated in the system and the system's context will use it.

1.4 The Application

To realize the design suggestion, a prototype was developed. The prototype was implemented in Macromedia's software Director MX. The inbuilt programming language in Director MX is Lingo.

Chapter 2

Task, Resources and Goal

To be able to do a well-planned project the task, resources and the goal have to be defined. In the following sections these are described further.

2.1 Task

Hägglunds have for decades built military vehicles. In recent years the vehicles have been equipped with many well developed computer systems. This project handles a fire control and information application in the vehicle-mounted grenade launcher system AMOS (Advanced MOrtar System). To inspect the GUI a prototype called WCS (Weapon Control System) has been developed. The purpose with this master thesis was to evaluate and further develop a suitable GUI for this fire control and information application known as MIS (Mortar Information System), which is a subset in VIS (Vehicle Information System), in a new design suggestion. The application was based on a former prototype, WCS, which had been tested by employees and recruits.

2.2 Resources

Input to this master thesis were results and data from an earlier master thesis [3], the WCS, a function tree which was updated by the users, comments from the users, a two-column display with touch-screen functionality, and contact with users and system developers. The software used to develop the prototype was Macromedia Director MX.

2.2.1 Preconditions

There were circumstances that had to be considered before starting the redesign session. For example the number of function buttons had to be reduced compared to the WCS. Therefore a challenge in the GUI redesign was to compress the functions into a fewer number of function keys. Another thing that had to be considered was that a physical keyboard is provided to the user instead of typing the numbers and letters from the GUI. In the WCS the numbers and letters were typed by the function buttons in the GUI, similar to writing a SMS from a cellular telephone. Besides this there were functionalities removed, compared to the WCS, which brought that function buttons could be removed.

2.3 Goal

The goal of this project is to make a design proposition which is implementable from the system developer's point of view and usable from the user's point of view. Another goal with this project was that the design proposition, to some extent, will be useful in the future system.

2.4 Work Procedure

When the task, resources and goal were stated the work process was decided. Below the work procedure is presented in a chronological order.

1. SGT
2. Heuristic evaluation
3. The first design proposition
4. User test
5. The second design proposition

2.5 Output

The results from this project is presented in two separate reports (one to Hägglunds and one to Umeå University), and two oral presentations (one to Hägglunds and one to Umeå University). Besides this, models, presentations, prototypes and such, created during the work period serve as output.

Chapter 3

Methods and Materials

This chapter provides a theory session and also describes methods used and how the project was performed.

To discover as many problems as possible in the prototype, a number of evaluation methods were used. This is called triangulation [23]. These methods have various ways of collecting comments on the prototype. Nielsen and Mack [17] suggest that a heuristic evaluation first should be done and later, after the redesign, do user testing since the two methods complement each other [15]. This was also done in this project.

However, the first method used was SGT (sub-goal template), which is predictive and models the user's behavior. The second evaluation method, heuristic evaluation, is also predictive but instead of modeling the user's behavior it uses interaction design experts which use their knowledge about usability to discover problems with the interface. The third and last method was user tests. User tests exploit an evaluation paradigm called usability testing and a category of an evaluation technique where users are asked about their opinion about the system. All these paradigms, techniques, and methods are described further in the following sections. A summary of the connection between the paradigms, techniques, and methods used in this study is shown in Table 3.1.

3.1 Evaluation Paradigms

Evaluations are lead by beliefs which can be strengthened by theory [23]. The beliefs together with their respectively techniques/methods is called evaluation paradigm.

In Table 3.2 four well-known evaluation paradigms are presented: "quick and dirty", field studies, predictive evaluation, and usability testing.

The evaluation paradigms used in this project are usability testing and predictive evaluation. Below these are explained further.

Table 3.1: A summary of the paradigms, techniques, and methods used.

	Paradigm	**Category of Technique**	**Method**
1	Predictive	Model user's behaviour	SGT
2	Predictive	Asking experts	Heuristic evaluation
3	Usability testing	Asking users	User test

3.1.1 Predictive Evaluation

In **predictive evaluation** the users do not have to be involved, instead experts use their knowledge of the typical user to find usability problems [23]. The experts are often guided by heuristics. Some advantages with predictive evaluations are that it is a quick and relatively inexpensive method because there is no need for users to be present in

Table 3.2: Characteristics of different evaluation paradigms. Based on Table 11.1 in [23, p. 344].

	Quick and dirty	**Field studies**	**Predictive**	**Usability testing**
Role of the user	Natural behaviour.	Natural behaviour.	Users generally not involved	To carry out set of tasks.
Who controls	Evaluators take minimum control.	Evaluators try to develop relationships with users.	Expert evaluators.	Evaluators strongly in control
When used	Any time you want to get feedback about a design quickly. Techniques from other evaluation paradigms can be used - e.g., experts review software.	Most often used early in design to check that users' needs are being met or to assess problems or design opportunities.	Expert reviews (often done by consultants) with a prototype, but can occur at any time. Models are used to assess specific aspects of a potential design.	With a prototype or product.
Type of data	Usually qualitative, informal descriptions.	Qualitative descriptions often accompanied with sketches, scenarios, quotes, other artifacts.	List of problems from expert reviews. Quantitative figures from model, e.g., how long it takes to perform a task using two designs.	Quantitative. Sometimes statistically validated. Users' opinions collected by questionnaires or interviews.
Fed back into design by...	Sketches, quotes, descriptive report.	Descriptions that include quotes, sketches, anecdotes, and sometimes time logs.	Reviewers provide a list of problems, often with suggested solutions. Times calculated from models are given to designers.	Report of performance measures, errors etc. Findings provide a benchmark for future versions.
Philosophy	User-centered, highly practical approach.	May be objective observation or ethnographic.	Practical heuristics and practitioner expertise underpin expert reviews. Theory underpins models.	Applied approach based on experimentation, i.e., usability engineering.

the process.

SGT is a predictive method which is done by evaluators. Often there are no users present but in this study the user's opinion indirectly were used in the SGT evaluation process. In the heuristic evaluation the users were not present either, but to some extent they influenced the heuristic evaluation since their comments, which were collected by Anundi and Holm [3], were considered.

3.1.2 Usability Testing

Usability testing is an important evaluation method, even though it had its peak already in the 1980s [28]. In usability testing the performance of typical users on typical tasks is measured [23]. Besides the performance tests, the users' satisfaction can be evaluated through questionnaires and interviews.

Usability testing is controlled by the evaluator [13]. Often the tests take place in a laboratory with no impact from the rest of the world and everything the test participant is doing is registered [23].

One main purpose with usability testing is to quantify users' performance [23]. Variables are not manipulated and the number of test participants is often too small for statistical analysis. The data from the user satisfaction test are analyzed and can then be summarized in a usability specification where the optimal performance levels and minimal levels of acceptance are specified and current levels are noted. The usability specification can serve as a reference when testing future versions of the system.

In this study usability testing was used in the user test. Users tested the first design suggestion and commented on it. The comments were collected in inquiries and oral discussions which after the test were assembled in a list of design suggestions from the users.

3.2 Evaluation Techniques

Preece, Rogers, and Sharp [23] discuss five categories of evaluation techniques: observing users, testing user's performance, modeling user's task performance, asking experts their opinions, and asking users their opinion to predict the efficiency of a user interface.

In this project evaluation techniques from three categories were used: modeling user's task performance, asking experts, and asking users. These are described further below. A presentation of how these evaluation techniques are connected to the paradigms mentioned above is also provided.

3.2.1 Model User's Task Performance

If there is no prototype to test and no users available, the user's task performance can be modeled [23]. This technique is most successful on systems with limited functionality, for example a telephone system or the system evaluated in this project. Within this category of techniques the keystroke model and GOMS (goals, operations, methods, and selection rules) are the most well-known. In this study the SGT method was used. More about SGT in Chapter 3.3.1.

Modeling user's task performance is only done in predictive evaluations [23]. There it is used to compare performance times of versions or predict the efficacy of an interface.

3.2.2 Ask Experts

By using heuristics, experts can identify problems with the GUI through a systematically search of the interface [23]. This is a relatively inexpensive method and quick, compared to techniques where users are involved. Therefore this technique is widely used by developers.

The "asking experts" category is only used in "quick and dirty" and in predictive evaluations [23]. In the former it is used to provide critiques ("crit reports") of the prototype's usability. In predictive evaluations heuristics are used early in the design to predict the efficiency of the interface.

In this study heuristic evaluation is used, which as mentioned earlier uses the "asking experts" technique.

3.2.3 Ask Users

Asking users of their opinion is a natural way of getting feedback from the users [23]. This category is very broad and can be used in a variety of ways; the number of participants can vary from just a few to hundreds, and either the questions are unstructured or tightly structured.

This category is also used in various ways in different evaluation paradigms [23]. In "quick and dirty" discussions with users take place. This is done either individually, in groups or in focus groups. User satisfaction questionnaires or interviews are used in usability testing. When using the "asking user" technique in field studies the evaluator interviews or discusses the situation with the participant. In predictive evaluation asking users are not applicable.

In the user test in this study the users were asked questions both written and oral. In the written session an inquiry were used and put together in a list with design suggestions from the users. The oral session served as a discussion forum, with the end users and the system designer, to find new design propositions.

3.3 Evaluation Methods

There are many ways to evaluate. In this project three different kinds of evaluations were used: SGT (sub goal templates) which is part of the task analysis family, heuristic evaluation, and user tests. SGT is used to model the user's performance and in heuristic evaluations experts use design guidelines to evaluate the system. In the user test used in this study the users' opinions about the system were registered both written and oral. Below the three methods are explained further. The connection between the theory and this specific project is provided in the "How?" sections.

According to Jeffries and Desurvire [10] the best way to evaluate a user interface is to adopt more than one evaluation method. In Nielsen's survey [16], he discovered that user testing and above all heuristic evaluation was the most successful method. User test is superior in the quality of the data it generates. Heuristic evaluation presents good results in that it generates good/useful information, the resources and time needed, the expertise/skills required, the interaction between multiple methods among others.

3.3.1 Task Analysis

What?

The term **task analysis** covers techniques for analyzing cognitive processes and physical actions [23]. This is done at a high level of abstraction and is also very precise and detailed. Task analysis is applied to test and examine an existing situation, not to develop new systems or artifacts. The output from a task analysis can be used to establish existing practices on which to create new requirements or to design new tasks. The method is used to investigate *how* people are acting, *why* they are doing what they are doing, and above all analyze the underlying *purpose* of what people are doing. Examples of task analysis techniques are HTA (hierarchical task analysis) and GOMS (goals, operations, methods, and selection rules).

A disadvantage with task analysis is that it does not scale well [23]. If the task has an enormous amount of subtasks it is impossible to cover them all. Another disadvantage with task analysis is that it does not handle all kinds of tasks, for example tasks that overlap or run in parallel. Besides this it cannot handle interruptions.

HTA Hierarchical task analysis (HTA) entails breaking tasks into subtasks and then the subtasks into sub-subtasks and so on [23]. To illustrate how the task is performed in an actual situation, the tasks and subtasks are grouped into plans. HTA does not look at actions related to software or an interaction device at all, instead the physical and observable action is considered.

If the hierarchy is deep, the information in the leaves might be irrelevant since it has the wrong focus. Therefore there is a need for a stop rule to know when to stop searching for subtasks. Annett and Duncan [2] introduced a rule called the *PxC criterion*. The rule is used to not make the HTA too analytical and to have the right focus [1]. The rule goes: "Stop when the product of the probability of failure (P) and the cost of failure (C) is judged acceptable." [1, p. 71-72].

HTA can be illustrated both in a graphical representation and with text, see Figure 3.1 and Figure 3.2 [23].

SGT Ormerod and Shepard [21] developed the sub-goal template (SGT) method to provide an approach for capturing system operation and user information requirements and at the same time minimize the input and maximize the quality of the output from the evaluation. The SGT method is similar to HTA but SGT is developed to be used in interface design.

According to Ormerod and Shepard [21], task analysis should model both existing and nonexisting user-system interactions. The authors claim that HTA has weaknesses in delivering a usable product to designers. The result of the SGT analysis is however user centered information requirements which are usable in the design process.

Instead of using the *PxC criterion* to know when to stop, SGT stops when the operation is an information-handling operation (IHO) that has to be handled outside the system, see Table 3.3 [21].

Besides IHO:s another feature that is different from HTA is the sequencing elements [21]. These sequencing elements are shown below in Table 3.4 and are used to construct plans. As mentioned before, it is hard to symbolize parallel processes in task analysis. In SGT this has been solved through a sequencing element that handles parallel processes. An example of a SGT for finding an article on the internet is presented in Figure 3.3 and 3.4.

Table 3.3: Subgoal templates (SGTs), specific task elements, and information requirements. Based on Table 17.2 in [21, p. 356].

SGTs	*Task Element*	*Context for Assigning SGT and Task Element*	*Information Requirements*
Act		Perform as part of a procedure or subsequent to a decision made about changing the system.	Action points and order; Current, alternative, and target states; preconditions, outcomes, dependencies; halting, recovery indicators.
	A1 Activate	Make subunit operational: switch from off to on.	Temporal/stage progression, outcome activation level.
	A2 Adjust	Regulate the rate of operation of a unit maintaining "on" state.	Rate of state change.
	A3 Deactivate	Make subunit nonoperational: switch from on to off.	Cessation descriptor.
Exchange		To fulfil a recording requirement. To obtain or deliver operating value.	Indication of item to be exchanged; channel for confirmation.
	E1 Enter	Record a value in a specified location.	Information range (continuous, discrete).
	E2 Extract	Obtain a value of a specified parameter.	Location of record for storage and retrieval; prompt for operator.
Navigate		To move to an informational state for exchange, action, or monitoring.	System/state structure, current relative location.
	N1 Locate	Find the location of a target value or control.	Target information, end location relative to start.
	N2 Move	Go to given location and search it.	Target location, directional descriptor.
	N3 Explore	Browse through a set of locations and values.	Current/next/previous item categories.
Monitor		To be aware of system states that determine need for navigation, exchange, and action.	Relevant items to monitor; record of when actions were taken; elapsed time from action to the present.
	M1 Monitor to detect deviance	Routinely compare system state against target state to determine need for action.	Normal parameters for comparison.
	M2 Monitor to anticipate cue	Compare system state against target state to determine readiness for known action.	Anticipated level.
	M3 Monitor transition	Routinely compare rate of change during state transition.	Template against which to compare observed parameters.

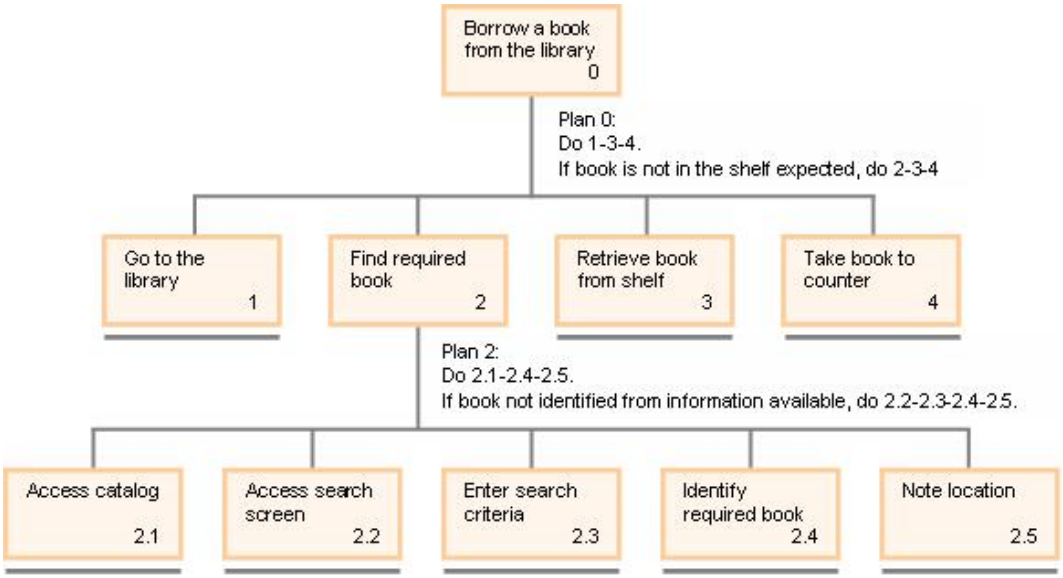

Figure 3.1: Graphical representation of HTA for borrowing a book from the library [23, p. 233].

0. In order to borrow a book from the library
   1. Go to the library
   2. Find the required book
      2.1. Access library catalog
      2.2. Access the search screen
      2.3. Enter search criteria
      2.4. Identify required book
      2.5. Note location
   3. Go to correct shelf and retrieve book
   4. Take the book to checkout counter

Plan 0: Do 1-3-4. If book is not on the shelf expected, do 2-3-4.
Plan 2: Do 2.1-2.4-2.5. If book not identified do 2.2-2.3-2.4-2.5.

Figure 3.2: Text-based HTA for borrowing a book from the library [23, p. 233].

Table 3.4: Sequencing elements used in SGT. Based on Table 13.3 in [21, p. 359].

Code	*Type*	*Syntax*
S1	Fixed sequence	S1 Do X...
S2	Contingent sequence	S2 If (c) then do X...
		If not (c) then do Y...
S3	Parallel sequence	S3 Do together
		X...
		Y...
S4	Free sequence	S4 In any order do
		X...
		Y...

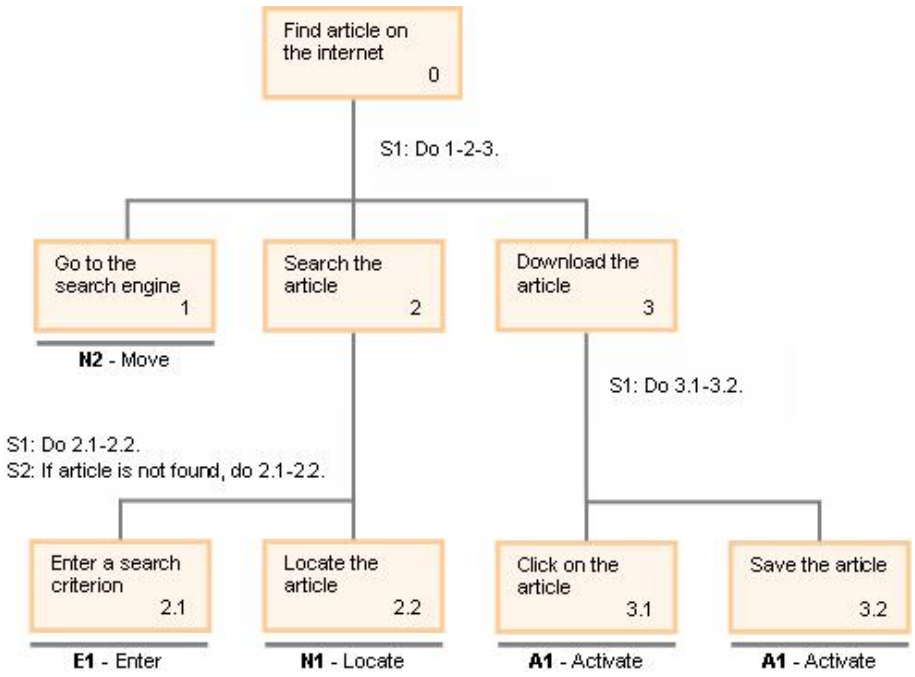

Figure 3.3: Graphical representation of SGT for finding an article on the internet.

0. In order to search and download an article on the internet
S1: Do 1-2-3.
1. Go to the search engine — **N2** - Move
2. Search the article
S1: Do 2.1-2.2.
S2: If article is not found, do 2.1-2.2.
2.1. Enter a search criterion — **E1** - Enter
2.2. Locate the article — **N1** - Locate
3. Download the article
S1: Do 3.1-3.2.
3.1. Click on the article — **A1** - Activate
3.2. Save the article — **A1** - Activate

Figure 3.4: Text-based SGT for finding an article on the internet.

How?

The first method used in this project was the SGT method. The SGT evaluation was made with the master thesis students Andreas Anundi and Anders Holm [3]. Since all three were novices in the area of combat vehicles we applied use cases, documents, and comments from users and staff at Hägglunds as input to understand the procedures in the vehicle. Even though there were a lot of information sources it was hard to get a deep understanding since the system is huge with an enormous amount of functions. Furthermore the procedures around the system are hard to understand without seeing them in practice. The use cases were not completed and were on another level than the SGT evaluation. Furthermore the use cases were too specific and not as general as the SGT required as input. An explanation of why the use cases were too specific could be

that they were made *after* the WCS prototype was developed. All this made it hard to synchronize the knowledge from the use cases to goals and sub-goals in the SGT.

3.3.2 Heuristic Evaluation

What?

Jakob Nielsen and his colleagues are the developers of the informal usability inspection technique called **heuristic evaluation** [17]. The method involves experts who are guided by a set of principles known as heuristics. The term heuristics is used to refer to design and usability principles when applied to a certain design problem. When design and usability principles are used in practice they are usually called heuristics.

Usability principles and design principles are approximately equal but usability principles tend to be more prescriptive, thus it does not represent the actual situation [23]. Furthermore design principles tend to be used mainly for informing a design and usability principles are often used for evaluating prototypes and existing systems. In the heuristic evaluation made in this project, both design and usability principles are used as guidelines to evaluate whether user-interface elements such as dialog boxes, menus, navigation structure, etc. match the principles. Below the principles are presented.

Design Principles Design principles are used as reminders of what to provide and what to avoid when designing an interface [23]. The most common design principles are visibility, feedback, constraints, mapping, consistency, and affordances [23]. Another design principle is simplicity, which is very important in website design according to Nielsen and Mack [17]. These design principles are presented below.

Visibility "The more visible functions are, the more likely users will be able to know what to do next" [23, p. 21]. Visibility is therefore one way to inform the users on what is happening [23].

Feedback Feedback is, according to Norman [19, p. 99], to "give each action an immediate and obvious effect.". Examples of types of feedback in interaction design are audio, tactile, verbal, visual, and combinations of these [23]. Since feedback is related to the concept of visibility, feedback can help provide the necessary visibility for the user interaction.

Constraints The concept of constraints in interaction design is to hide actions that are not allowed to execute.

Norman [20] classifies constraints into three categories: physical, logical, and cultural.

Physical constraints - The way physical objects restrict the movement of things.

- E.g., a physical button is to be pushed and not pulled.

Logical constraints - The way people understand how the world works.

- E.g., disabling menu options when they are not appropriate to execute, see Figure 3.5.

Cultural constraints - In learned conventions.

- E.g., the color red means "warning" and the smiley :-) means "happy".

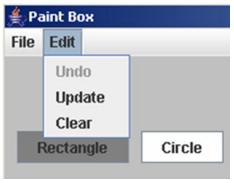

Figure 3.5: Example of a logical constraint. The Undo action is inactivated because it is not possible/allowed to perform the Undo action.

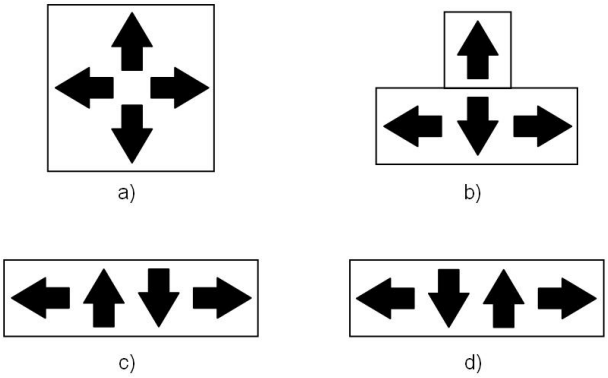

Figure 3.6: Example of mapping. Four different placements of controls. Based on Figure 1.10 in [23, p. 23].

Mapping Mapping concerns the relationship between controls and their effects in the world [23]. The controls should be placed in an established way that is accepted by the user. In Figure 3.6 four different placements of the controls are shown. Figure 3.6d is preferred before 3.6c.

Consistency Consistency means having similar operations and use similar elements for achieving similar tasks [23]. One advantage with consistent interfaces is that they are easier to learn and use. The way operations are grouped into categories should also be consistent. For example it is logical that the file handling operations appear in the "File" menu in the menu bar. One question is if the virtual world should be consistent with the physical, e.g., how to lock items.

Affordances Norman [19, p. 9] defines affordances as: "the perceived and actual properties of a thing, primarily those fundamental properties that determine just how the thing could possibly be used." It is the attribute of an object that helps users distinguish what to do with the object [23].

Norman [20] claims that there are two types of affordances: perceived and real. Real affordances appear on physical objects, for instance by grasping which is perceptually obvious and not learned. Perceived affordances appear in user interfaces that are screen-based, in other words they are not real but virtual and have to be taught. Norman claims that it is not appropriate to try to design for real affordances at the graphical interface. Norman argues that design concepts like conventions, feedback and cultural and logical constraints are more useful in designing an interface than affordances.

Simplicity In website design, simplicity is an important design principle according to Nielsen and Mack [17]. Sometimes it is better to remove design elements such as icons, buttons, boxes, lines, graphics, shading, and text, to make a cleaner and crisper interface and make it easier to navigate the website [23]. However, some "unnecessary" graphics are not appropriate to be removed since it can make the website more aesthetically attractive and therefore more easy to navigate and read. Therefore the most important thing is to find a balance between aesthetic appeal and the right type and quantity of information on the website.

Usability Principles Usability principles are used to assess the acceptability of interfaces [23]. The original set of heuristics was a result of a study where 249 usability problems were tested [15]. The ten main usability principles, according to Nielsen and Mack [17] are stated below. Questions that are generated from the statements are also provided.

1. **Visibility of system status** Always keep users informed about what is going on, through providing appropriate feedback within reasonable time.
   - Q1: Are users kept informed about what is going on?
   - Q2: Is appropriate feedback provided about a user's action?
2. **Match between system and the real world** Speak the user's language, using words, phrases and concepts familiar to the user, rather than system-oriented terms.
   - Q1: Is the language used in the interface simple?
   - Q2: Are the words, phrases and concepts used familiar to the user?
3. **User control and freedom** Provide ways of allowing users to easily escape from states they unexpectedly find themselves, by using clearly marked "emergency exits".
   - Q1: Are there ways of allowing users to easily escape from places they unexpectedly find themselves?
4. **Consistency and standards** Avoid making users wonder whether different words, situations, or actions mean the same thing.
   - Q1: Are there ways of performing similar actions consistent?
5. **Error prevention** Where possible prevent errors occurring in the first place.
   - Q1: Is it easy to make errors?
   - Q2: If so where and why?
6. **Recognition rather than recall** Make objects, actions, and options visible.
   - Q1: Are objects, actions and options always visible?
7. **Flexibility and efficiency of use** Provide accelerators (i.e., shortcuts) that are invisible to novice users, but allow more experienced users to carry out tasks more quickly.

   - Q1: Have accelerators been provided that allows more experienced users to carry out tasks more quickly?

8. **Aesthetic and minimalist design** Avoid using information that is irrelevant or rarely needed.
   - Q1: Is any unnecessary and irrelevant information provided?

9. **Help users recognize, diagnose, and recover from errors** Use plain language to describe the nature of the problem and suggest a way of solving it.
   - Q1: Are error messages helpful?
   - Q2: Do they use plain language to describe the nature of the problem and suggest a way of solving it?

10. **Help and documentation** Provide information that can be easily searched and provides help in a set of concrete steps that can easily be followed.
    - Q1: Is help information provided that can be easily searched and easily followed?

Usability According to Preece et al. [23, p. 14] usability means "ensuring that interactive products are easy to learn, effective to use, and enjoyable from the user's perspective.". Below six usability goals are presented [23].

- Effective to use (**effectiveness**) - How good a system is at what it is doing.
  - Q: Is the system capable of allowing people to learn well, carry out their work efficiently, access the information they need, buy the goods they want etc.
- Efficient to use (**efficiency**) - The way a system supports users to manage their tasks.
  - Q: Once users have learned how to use a system to carry out their tasks, can they sustain a high level of productivity?
- Safe to use (**safety**) - Protecting the user from dangerous conditions (e.g. X-ray machines) and undesirable situations (e.g. accidentally erase all e-mails in the inbox).
  - Q: Does the system prevent users from making serious errors and, if they do make an error, does it permit them to recover at all?
- Have a good utility (**utility**) - The extent to which the system provides the right kind of functionality so that users can do what they need or want to do.
  - Q: Does the system provide an appropriate set of functions that enables users to carry out all their tasks in the way they want to do them?
- Easy to learn (**learnability**) - How easy the system is to learn to use.
  - Q: How easy is it and how long does it take (i) to get started using a system to perform core tasks and (ii) to learn the range of operations to perform a wider set of tasks?

– Easy to remember how to use (**memorability**) - How easy a system is to remember how to use once learned.

  - Q: What kinds of interface support have been provided to help users remember how to carry out tasks, especially for systems and operations that are used infrequently?

Who?

To apply a heuristic evaluation, expert evaluators test the product and note the problems they come across [23]. Empirical studies suggest that 75 percent of all usability problems are usually identified if [three to] five evaluators [18] collaborate in the task [17]. However, many usability problems can be captured by a single evaluator expert, and therefore this technique is frequently used for critiquing interactive devices. This is called an expert crit, but also discount evaluation since users and special facilities are not needed and therefore it is comparatively inexpensive and quick.

For an expert to become an evaluator the estimated time needed to train/practice is one week, however this depends on the person's expertise [17]. The best experts will have expertise in both interaction design and the product domain, see also the summary of Nielsen's study [14] below. Even though there are claims that it is not successful, the typical user can also be taught to do heuristic evaluation [17]. According to Jeffries, Miller, Wharton, and Uyeda [11], software engineers should not evaluate the usability of a product since usability experts can find three times as many problems as the engineers and more than twice as many of the most important problems.

A "novice" evaluator has no training or experience in usability engineering [14], see Table 3.5. In Nielsen's survey [14] the "novice" evaluator found 22 percent of the usability problems. A "regular" usability specialist is an expert in usability engineering but has no expertise in the usability of the type of interface that is evaluated. The "regular" usability specialist found 41 percent of the usability problems in Nielsen's study. A "double" usability specialist is expert in usability engineering and also has experience working with the system. 60 percent of the usability problems in Nielsen's survey were found by what he called "double" usability specialists. According to these definitions I am a "regular"-"semi-double" usability specialist since I have a "usability" education and have some knowledge of the system.

Nielsen's study also revealed that a group of five novices found 51 percent of the problems, five regular usability specialists found 80 percent and five double usability specialists found 98 percent of the problems [14].

Bailey's study [4] supports the use of human factors specialists in user interface design. He found significant difference between designs produced by human factors specialists and programmers.

Why?

Jeffries et al. [11] compared four evaluation methods in their effectiveness in discovering usability problems. The four evaluation methods were applying guidelines, heuristic evaluation, cognitive walkthroughs and usability test. With applying guidelines Jeffries et al. refer to a method where software engineers were served guidelines of good practice in usability and studied them. Then they used the guidelines to evaluate the user interface. Cognitive walkthroughs is when a group of software engineers together walks through the user interface and identifies usability problems. The study revealed that

the heuristic evaluation found 105 problems, applying guidelines 35, the walkthrough 35, and the usability test 31. However the usability test found more severe problems then the guidelines and the walkthrough. Usability testing finds more global problems and heuristic evaluation finds more local ones. Comparing the cost of each evaluation method the heuristic evaluation only took 1/10 of the time it took to conclude the usability testing, and therefore the heuristic evaluation brings the highest cost-benefit.

However, Desurvire, Kondziela, and Atwood [7] discovered that the usability test found more than twice as many problems as the heuristic evaluation. Dumas and Redish [8] recommend to do both usability testing and heuristic evaluation.

Critique of Heuristic Evaluation According to Dumas and Redish [8], Bailey, Allan, and Raiello [5] showed that repairing many of the local problems found by heuristic evaluation did not improve the usability of the interface.

How?

In heuristic evaluations three to five interaction design experts are preferable [18]. Since this project has a tight time table and a low budget, the heuristic evaluation had to be reduced to involve only one expert - the author herself. To be able to do the heuristic evaluation, information about the system and how it is used had to be collected. The SGT evaluation and comments from users and staff at Hägglunds were important information sources. The comments from the users were collected by Anundi and Holm in their master thesis project [3].

The heuristic evaluation was put together with the design and usability principles mentioned earlier. The outcome was a list with both explanations on why a certain principle was important to consider and questions to answer on how the principle was used in the interface. The interaction design expert followed the list and answered the questions about the usability of the interface. The list was then put together into another list with new design suggestions. This design suggestion, together with the result/design suggestions from the SGT evaluation, was realized in a prototype.

Table 3.5: A summarize of Nielsen's survey [14].

	Novice	**Regular**	**Double**
Experience in working with the interface	No	No	Yes
Experience in usability engineering	No	Yes	Yes
Findings (one evaluator)	22%	41%	60%
Findings (five evaluators)	51%	80%	98%

3.3.3 User Tests

What?

User testing and scientific experiments have much in common since both measure performance [23]. However there is a difference; user testing is a systematic approach to evaluate user performance used to inform and improve usability design while scientific experiments discover new knowledge. The results from these two methods also differ. Experiments use statistical tests and user tests, if they use statistics, only involve means and standard deviations.

Bailey [4] revealed three difficulties that have to be handled after a user test. The difficulties were. . . :

1. . . . to recognize usability problems based on the comments since the designer must be able to separate actual usability problems from comments due to individual differences.

2. . . . to figure out a way to find a way to repair the problem when a problem is identified.

3. . . . to try to take a weak design and attempt to reach a "quality" design through iteration.

Who?

According to Dumas and Redish [8] a user test typically involve 5-12 users. However, fewer test participants are often used to match budget and time schedule. This study included five users; two actual end users, one internal tester at Hägglunds, and two internal system developers at Hägglunds. An important criterion when choosing the test participants in this project was that they should be familiar with the system and how it is used. The purpose of using system developers in the test was to make sure the design proposition was implementable. In the same way the end users were important to secure usability.

Why?

The user's opinion about the system is very useful, because if the users do not like the system they will not use it [23], even if they sometimes have to.

User tests are closely related to usability testing. Usability testing will find global problems with the interface and also problems that other methods miss [8]. However, if only doing a user test there is a risk of missing many local problems. By missing many local problems the product will feel negligent.

How?

To test if the design suggestion from the SGT and heuristic evaluation appeal to the users, a user test was made. To collect the user's opinion questionnaires and interviews were made. The written part had an information section and a comment section. The information section included a brief manual for the users to read before testing the prototype. After completing the information section the test person performed seventeen tasks in the prototype and commented on them. When completing these tasks the test

person examined the menu structure in a hierarchical menu tree and commented on it. At the end of the test the test person gave his/her overall comments on the system.

The written part of the user test was made without any supervisor present to collect oral spontaneous comments. To be able to collect these comments an oral session was also made. In the oral part the test persons and the developer had the opportunity to ask each other questions and give more comments and together make a new design proposition.

The result from the user test was then visualized in a second design suggestion which was also realized in a prototype.

Pilot Test To make sure the evaluation plan is executable a pilot test can be made [23]. A pilot test is a small pretest to make sure that all equipment, questions, instructions, procedure and so on is working as it should. When it is difficult to find test participants, colleagues or peers can be used to comment the test and its contents. Dumas and Redish [8] suggest that the pilot test should be done at least two days before the actual test. The days between the pilot test and the real test should be scheduled to make corrections and refinements in the test.

To "test the test", a pilot study was made on the written part. The pilot study involved the project's supervisor at Hägglunds. The pilot study revealed minor unclearness in the instructions and also language problems.

Iterative Design Iterative design is a design process done in cycles [27]. In each cycle the design is elaborated, refined, and tested. The result from the testing in each cycle is used in the next cycle. The design-prototype-test cycle is presented in Figure 3.7.

Bailey [4] showed that an iterative design methodology can improve the usability of a product.

The iterative design process has been used to some extent in this project since the user's comments on the early prototype was input to the first new prototype which were tested by the users. Later a second redesign was made from the comments from the user test.

3.4 Prototyping

What?

Prototypes are used as a communication device between the producer and stakeholders [23]. It is a limited representation of the design and allows users to explore and interact with the system. Prototypes can for instance be static paper-based or interactive software-based [8]. In this study the users tested an interactive software-based prototype since the system is rather complex and hard to illustrate in a low fidelity prototype.

Why?

One benefit by using prototypes is to test the system before it is too late to make changes [8]. Another advantage with interactive prototyping is if the system is complex it is hard to make an understandable static paper-based prototype. Other advantages with software prototypes are that they can bring feedback to the project early in the development process, many design suggestions can be made and tested, they make it

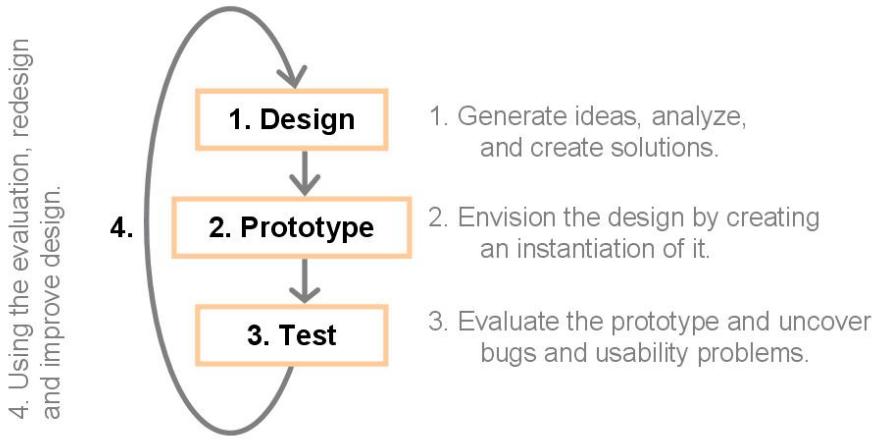

Figure 3.7: The design-prototype-test cycle [26].

possible to use iterative design, and make good communication materials. Besides this, a prototype is a good reflection device for the system developers in their design work [25]. Liddle [12] suggest that a prototype should always precede any writing of code.

Objections of High-Fidelity Prototyping There are many advantages with interactive software-based prototypes, but there are some objections to high-fidelity prototyping. Marc Rettig [24] advocate that low-fidelity prototypes should be used in more projects because high-fidelity prototypes bring problems such as:

- Takes too long to build.
- Testers tend to comment on shallow details instead of the content.
- Developers do not like to make changes in their prototypes which they have put many hours in.
- There is a risk that high-fidelity prototypes set too high expectations.
- If there is a bug in the prototype, the test may be disturbed.

How?

The prototype was made in Macromedia Director MX with the programming language Lingo. It was not possible to implement the whole menu structure and some of the functions for different reasons. Even though everything was not implemented the work with the prototype was time severe and lasted about six weeks, which is 1/3 of the whole project. The result was a semi-functional interactive system to serve as a visualization of the design suggestion to the testers.

3.5 The DECIDE Framework

The framework known as DECIDE helps the evaluator plan the evaluation study [23]. The DECIDE framework is a help to carry out a well-made evaluation. The framework is an acrostic list and will be further described in the following sections.

1. **D**etermine the overall *goals* that the evaluation addresses.
2. **E**xplore the specific *questions* to be answered.
3. **C**hoose the *evaluation paradigm* and *techniques* to answer the questions.
4. **I**dentify the *practical issues* that must be addressed, such as selecting participants.
5. **D**ecide how to deal with the *ethical issues*.
6. **E**valuate, interpret, and present the *data*.

3.5.1 Determine the Overall Goals

This is an important part of the evaluation process since the goals guide it all the way to the end [23]. This section has the power to clarify the user's needs, find a suitable metaphor for conceptual design, revise the interface, etc. It can also identify who wants the evaluation and why.

The main goal with this project is to improve the GUI from a HCI perspective. Besides this the users should have a natural mental model of the system. The user's model is provided by the system's image which the designer has to create through his/her own mental image.

Mental Model

"[The mental model is] the model people have about themselves, others, the environment, and the things with which they interact." [19, p. 17]. The mental model is formed through experience, training, and instruction. An illustration of mental model is presented in Figure 3.8.

Design model - The designer's conceptual model.

User's model - The mental model developed through interaction with the system.

System image - The visible part of the device that has been built.

According to Norman [19] the designer expects the user's model to be identical to the design model. When the product/system is completed there is no direct communication between the designer and the user, all communication is through the system image. This puts a lot of pressure on the designer to give the system the "correct" image to be interpreted by the user.

According to Payne [22], mental models seem to contribute more to the area of HCI if the focus is on the mental content instead of structural aspects such as the cognitive architecture. Besides this, mental models should be built from instructions and the most common method is to use fragmentary models of the systems used.

To help the users easily navigate in the prototype provided in this project, different types of buttons have been introduced, see Figure 3.9. These buttons should symbolize what happens if they are pressed, which gives a (hopefully) good system image and by that a correct mental model to the user.

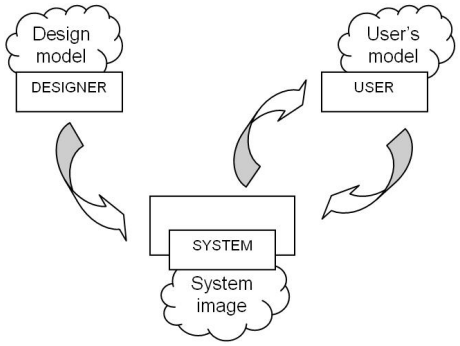

Figure 3.8: Illustration of the relation between the designer's model, the user's mental model and the image the system presents. Based on Figure 1.10 in [19, p. 16].

3.5.2 Explore the Questions

To be able to operate the goals many questions have to be answered [23]. To make the evaluation more specific the questions can be decomposed into sub-questions. For example: the question "Is the user interface bad?" can be divided into questions such as: "Is the system difficult to navigate?", "Is the terminology confusing?", etc. These sub-questions can also be further refined.

The preparatory work to the heuristic evaluation served as a question finder in this project. Questions like "Is the interface usable?" and "Is the interface providing a good mental model for the user?" were found.

3.5.3 Choose the Evaluation Paradigm and Techniques

After discovering the goals and questions, the evaluation paradigm and technique has to be chosen [23]. The evaluation technique is dependent on the paradigm chosen. Besides this, issues such as ethics and how to practically do the evaluation are considered. If the chosen technique for example is expensive, takes time or requires equipment that is not available, then the technique has to be reconsidered and compromises has to be made.

The methods that are used in this project are SGT, heuristic evaluation and user testing. The heuristic evaluation was relatively informal with one expert evaluator. After the SGT and heuristic evaluation there was a redesign of the interface which was evaluated by the users.

3.5.4 Identify the Practical Issues

Issues that include users, facilities and equipment, schedules and budgets, and evaluators' expertise has to be considered *before* starting [23].

When it is time to choose the users, there are many things to consider [23]. Examples of things to regard is the number of men and women to include, what kind of type should the users be (novices, experts, the ones in the middle, or a mix of them), what kind of tasks should be executed and how long should each task be scheduled. The evaluators have to make the participants comfortable so they do not feel anxious and perform as usual. This can be done by explaining that the users are not being tested, the system is.

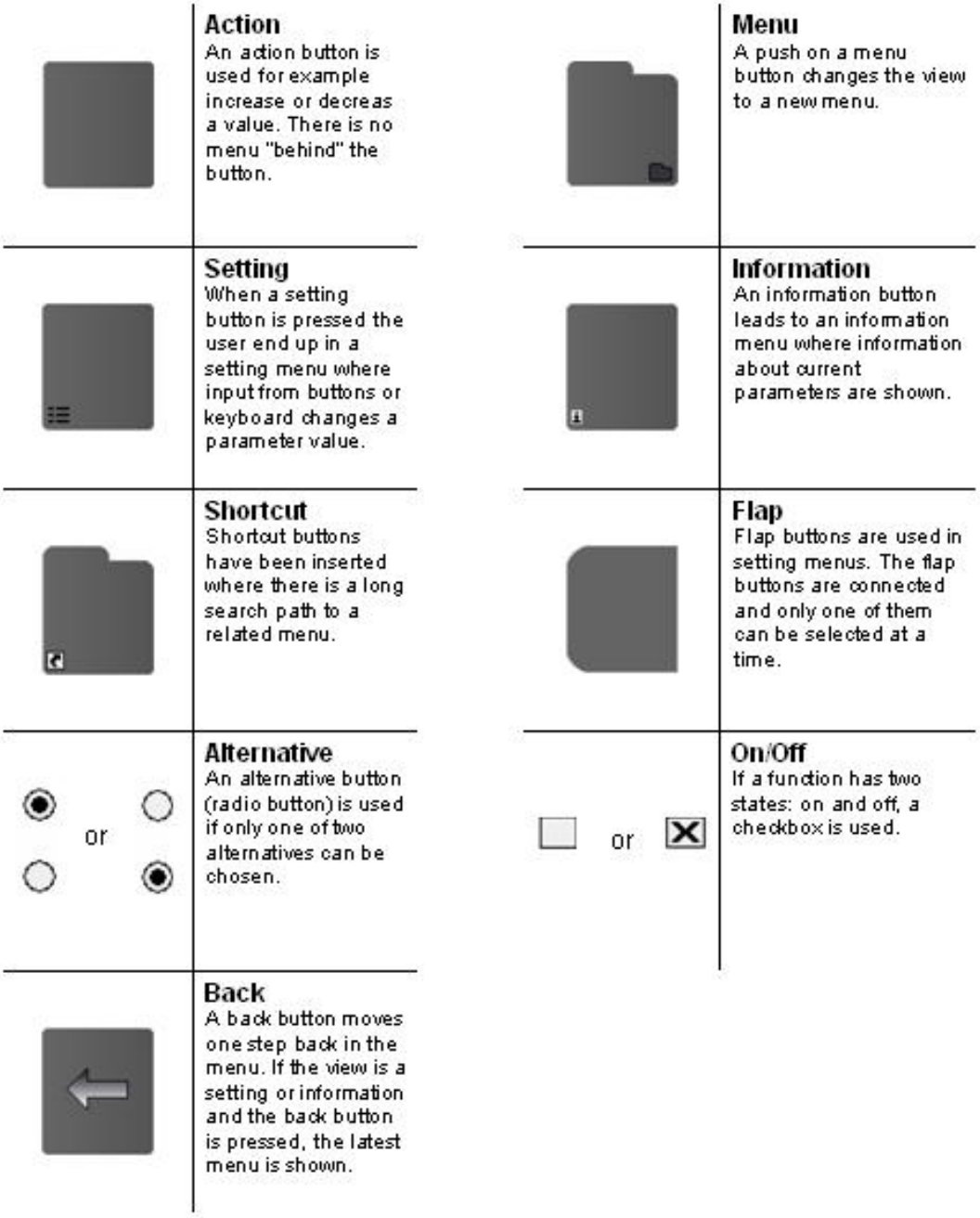

Figure 3.9: The different types of buttons in the prototype.

When it comes to facilities and equipment there are many questions that need to be considered [23]. To be sure nothing goes wrong, spare equipment is a good back-up. Time and budget constraints have to be reflected upon [6]. The evaluators have to compromise and plan to do a good job with the resources and time available [23].

The evaluators need to reflect on whether they have the right expertise to perform the evaluation as planed. If the results are to be statistically analyzed a statistician is to be preferred.

Most of the practical issues were not yet decided when entering the evaluation part of the project. The most critical part in the project was whether the most suitable users were available in the user test. However this worked out well, but at least one week was lost during this discussion. Even though everything was thoroughly planned, the less time it was until the user test more ideas and circumstances had to be included which brought rearrangements in the plan.

The users that were chosen were persons that were familiar with the former system. Before entering the test the test participants were informed that it was not them that were tested, the system was. To make sure all equipment worked as expected and that the test was constructed well, a pilot test was made.

3.5.5 Decide How to Deal With the Ethical Issues

The participant's privacy should be respected and protected [23]. Health status, employment, education, financial status, and where the participants live should be confidential [23].

Preece et al. [23] have stated some guidelines to help ensure that evaluations are done ethically and that the participant's rights are respected:

1. Be open to the participants of goals with the evaluation and what the procedure looks like.
2. Make sure that the participant understands that all personal information, such as financial, address, health, will be confidential.
3. Make sure the participants are well understood that they can abort the evaluation at any time.
4. If possible give the participant a payment because then the relationship between the participants and the evaluators will be more formal.
5. When referring to a participant in the report, one should not be able to identify the person.
6. If it is necessary to quote a participant, permission has to be given to do that. It can be appropriate to promise them anonymity and offer them a copy of the report.

At the beginning of the user test session the participants received an explanation of the background, the procedure and the aims of the survey. It was not possible to give the participants any payment. To be able to separate the participants in the analysis part of the process, identification names were used: TP1 (test person 1), TP2, TP3 and so on. However, in the report no names or anything else that could identify the participants was used, which the participants were notified before the test started.

3.5.6 Evaluate, Interpret, and Present the Data

After doing the user test the result had to be analyzed. There are many ways of doing that and it depends on the appearance of the results, the technique used. But there are, however, some issues that are always important when choosing technique [9].

1. The technique should be **reliable** in that it produces the *same* results on separate occasions with separate evaluators under the *same* circumstances and *same* procedure.

2. The **validity** of the technique is important. That is that the evaluation technique should measure what it is supposed to measure. This depends on both the technique and the persons performing the evaluation.

3. **Biases** occur when the results are unclear. An example of that is in heuristic evaluations where the expert evaluators are more sensible to one type of faults than others.

4. Different evaluation techniques have different **scope**, which is how much the result can be generalized.

5. The **ecological validity** of an evaluation is concerning how the environment has affected the results.

The SGT evaluation was to some extent not reliable. The evaluators could change the level of the approach from one day to another. That was a direct result from the, in this aspect, poor input to the evaluation. Besides this the evaluation technique did not give the result wanted. The work and preparatory work of the study was much more educational than the result itself.

In the heuristic evaluation the reliability should be strong since the evaluator is an expert. The result from the heuristic evaluation was useful and easy to work with when doing the design suggestion. However, the evaluator was probably more alert on the graphical problems than on the structural problems since her experience with the system was theoretical and not practical.

To make the user test reliable and valid it had to be carefully planed. Since the participants had dissimilar experience and education in the interface there may have occurred biases in that they are using the interface in different ways. Since the user test was performed with a small user group the scope was hard to generalize but it had to be done anyway. Hopefully the participants were experienced enough with the system and doing such tests that they were not affected by the environment and the circumstances.

Chapter 4

Results

Below the results of the three evaluation methods; SGT, heuristic evaluation, and user test; are presented. Besides the presentation of the evaluation results a description of the result from the two design sessions is presented. However, there are parts that are company confidential and will not be presented in this report.

4.1 SGT

When deciding to use the SGT method the intention was to learn more about the system and how it was used. But the SGT provided a pervading list of functions in the system and the level of the SGT did not make the evaluation result much useful in the design work. The preparatory work of the SGT was more useful than the actual result.

The SGT evaluation was based on use cases. These were not as goal driven as they should have been for this purpose. An explanation could be that the use cases were created *after* developing the WCS prototype. Therefore the WCS might have influenced the use cases. Hence the result of the SGT was too specific in the sense of not being applicable on any type of system with the same tasks.

The knowledge of the system, gained in the preparatory work with the SGT, was on how the system was used. Besides this, many of the system's tasks were identified, which were great help in the following evaluation and redesign sessions. However, since the system had a huge amount of functions, all could not be identified. The way the system was used was not either clearly specified. This type of information could easier be provided through field studies.

Since there is confidential information in the SGT result, it is not presented in this master thesis report.

4.2 Heuristic Evaluation

The heuristic evaluation of the WCS resulted in many suggestions of how to change the interface. Below examples of the result from the heuristic evaluation, that is the problems found in the WCS, are presented.

4.2.1 Design Principles

This section provides examples of the problems found in the heuristic evaluation that contradicts the design principled.

- **Visibility** The functions in the WCS do not guide the user by informing on what to do next.
- **Feedback** Some settings can be deleted without any feedback. Besides this, there are buttons not giving any feedback when pressed.
- **Constraints** Buttons that should not be pressed are marked with grey text. However even if there is not a function related to a button space, a button (without name) is visible.
- **Mapping** The left and right controls that handle the setting marker are on the same side in the interface.
- **Consistency** All buttons, independently of their function, look the same.
- **Affordances** As stated above, the buttons are not visualizing what they "do".
- **Simplicity** The WCS has simple graphics.

4.2.2 Usability Principles

This section presents a selection of problems, found in the heuristic evaluation, connected to the usability principled.

- **Visibility of system status** The location of the current menu in the system is not presented to the user.
- **Match between system and the real world** There was a button name that was shortened which gave the function another meaning.
- **User control and freedom** If settings are not set properly the user can be stuck in the setting menu. This is most likely to happen if the user does not have deep system knowledge.
- **Consistency and standards** Some functions have the same name but mean different things.
- **Error prevention** The evaluator hit setting errors several times when testing the WCS since the settings were not accurately set. If the user is unfamiliar with the appearance of the settings the only way to exiting the menu is to restart the prototype.
- **Recognition rather than recall** It is not visible that some buttons have menus under them. Furthermore some buttons are connected in that only one of them can be selected at a time.
- **Flexibility and efficiency of use** There are no accelerators provided.
- **Aesthetic and minimalists design** In one menu the user is always asked if he/she wants to remove an object even if there is no object.

– **Help users recognize, diagnose, and recover from errors** The evaluator only hit setting errors and there were no error messages presented there.

– **Help and documentation** There is no help provided.

Usability Goals

Problems found in the heuristic evaluation that contradict the usability goals are presented below.

– **Effectiveness** The system has a good effectiveness in that it provides the functions that it should.

– **Efficiency** Even though the evaluator made many attempts, the WCS was hard to learn.

– **Safety** The evaluation did not find any serious errors to prevent.

– **Utility** The system provides the functions it should.

– **Learnability** It is hard to enter numbers and there is no logical way of placing them.

– **Memorability** Even after many hours with the system it was hard to memorize the procedures and the location of functions.

4.2.3 A Template for Heuristic Evaluation

Another result, which was developed during the theory session, is a template for heuristic evaluation, see Appendix A. By using the template, a heuristic evaluation can be made internally at Hägglunds by their own staff. The template is adapted to the type of systems Hägglunds provides and helps the evaluator find important and relevant issues in the interaction design. Besides this, the heuristic evaluation can motivate the chosen design, which can be profitable from a marketing point of view.

4.3 The First Design Proposition

Since the users will get a thorough course in handling the system, it is important to remember that all actions do not have to be visualized and obvious at first sight of the system. The most important thing is that the functions are easy to remember once learned (memorability). In the following sections the changes and the functions that are added and removed in the interface are presented.

4.3.1 Changes

Some of the menu names in the WCS were ambiguous or too long and had to be changed in the new design proposition. Other modification was that colors which indicated function status were changed. For example, the status symbols were only marked red if there was an error. In the WCS the color red, besides indicating errors, shows that there exist values that are not accurately set. When the user does not have to concern about errors, the status icon, according to the heuristic evaluation, should be camouflaged with

the color grey, earlier it was marked green. A consequence from the heuristic evaluation was that the message icon turns green if there is an incoming message. Otherwise the color green is not used. When the system alerts the user, for example if a setting is inaccurate, the icon should be yellow and when an error occurs the icon turns red.

The overall structure of the system was considered since it had flaws in the logical and practical parts. Besides this, the reduction of the number of buttons made the rearrangement of the menu structure necessary. The menus had to be narrower and deeper to fit all functions. The result of restructuring the menus was intended to make the system easier to learn and remember.

In the WCS the popup menus were placed at the top of the interface on top of other relevant information. This was changed since the lower part of the interface had no relevant information and could be spared to contain the popup information. This is, however, the opposite of what most HCI literature suggests, since the top of the screen is seen to be the most important area of attention and therefore is a good place for a popup menu to be placed. Some of the related buttons were placed far from each other in the WCS, which to some extent was changed in the first design suggestion. However, there were some structural and practical issues that made some related buttons to be separated. For instance the up and down buttons had to be placed on one side of the interface and the left and right buttons on the other side since there were not four free button spaces on the same side.

4.3.2 Additions

The heuristic evaluation revealed that it was not obvious if the values just set are saved when exiting the menu. Therefore a “save changes” button was added. Whenever a critical action is performed, feedback is provided since the user should be updated on what is happening all the time. Besides this, actions that are not allowed or not possible to perform are locked and hidden.

In the WCS the evaluator was caught in so called dead-ends since she did not have knowledge of the parameters’ appearance. When there is a dead-end the user should have an opportunity to get information about what is missing/wrong to be able to change the parameters and move on.

The design suggestion introduced different types of buttons to facilitate the navigation, see Figure 3.9. The interface will be more usable by visualizing buttons that are connected, in that only one of them can be chosen at the same time. So called accelerators, i.e. shortcuts, were also provided which are used by experts to faster navigate the system.

Another help for the user that is applied in the design proposition is a headline on each menu with the name of the menu that the user is visiting. This is shown as a search path and will help in the navigation since it is easy to know the way that is traversed to get to current menu.

Another feature implemented in the first design proposition was a static button which had a direct connection with a system known as VIS. This button was visible in all menus.

4.3.3 Removals

Apart from the buttons being removed for not being usable in the WCS, buttons that were used for entering numbers and letters were removed since a keyboard is to be used

in the final system. Furthermore a confirmation question, that asked if an object should be erased even though there were no objects available, was removed in the cases where there were no objects.

4.4 User Test

The user test revealed that many of the propositions in the first design were good, but there were also a lot of misunderstandings in the design. An example of that is that the change of the colors on the status symbols was not functioning. The users liked that, in the WCS, the color red marked an error - any type of error that makes the system not working. The oral session where only the end users and the evaluator were present, discovered several new design alternatives that were useful in the latest design proposition. Below some hand-picked results from the questionnaire and the interview are presented.

4.4.1 The Questionnaire

The questionnaire in the user test showed that the users liked the design suggestion. Elements they expressed they fancied were the disposition, the navigation, the touch screen shortcut buttons, the search path, and the various types of buttons. However there were elements that were not welcomed by the users: the menu structure, the colors in the status field, the “save changes” button, the “undo” button, and the prototype’s degree of maturity.

4.4.2 The Interview

In the interview with the end users many problem areas, that were not discussed earlier, were discovered. Above all the focus on the map in the interface had been exaggerated since it was rarely used the way the system developers had in mind. With the map hidden a cleaner and simpler interface is provided.

Another result of the discussion with the end users was a total reorganization of the menu structure. It was useful to design the menu structure with the users since they know the actual work procedures. The result was a more efficient interface. Besides rearranging the menu structure, a redesign of the status symbols was made to better visualize the function.

Another comment from the end users was that a certain menu should not exist at all in the interface. This was because the function was also used in other platforms and therefore the function should be linked to the base system so the same graphical view is shown to the users independently of the platform used. The reason for this was to create a more consistent environment for the users.

An important comment from the users was that the fire control and information application did not have to be connected all the time with the VIS system. The users also asked for a higher degree of automatic handling of radio-messages.

4.5 The Second Design Proposition

Most of the second redesign was made in the user test interview. The focus was mostly on the menu structure in that session. However, some graphical design improvements

Figure 4.1: Illustration of the save symbol.

were also made. Below the changes, additions, and removals from the second and last design proposition are presented.

4.5.1 Changes

The menu structure was rearranged as a result of the user test interview session. Furthermore the information presented in the GUI was reconsidered and therefore a redesign of the GUI was made. The status symbols were also redesigned in the user test interview.

4.5.2 Additions

Since the users did not like the "save changes" button a new suggestion of that has been made. The users mean that it is enough to use the back button in the setting menus to save the settings. However this can be a bit confusing and therefore a "save symbol" has been added to visualize that saving occurs when the back button is used, see Figure 4.1.

A status symbol for navigation was added to the interface. Besides that, a function that hides the map was added.

4.5.3 Removals

The VIS system does not have to be reachable from all menus. This resulted in the static button reachable from all menus to be removed and therefore an extra button space was free to use for other purposes.

One of the status symbols was removed since the information it declared was not relevant.

Chapter 5

Conclusions

It was useful to apply three evaluation methods. It was also successful to mix two totally different methods before doing the first design suggestion. The SGT method, where there was no need to have an existing product, and the heuristic evaluation, where the existing prototype was tested, complemented each other. By doing the SGT I learned much about the system and how it is used and by using heuristic evaluation, features from the WCS could be exploited in the new design suggestion.

However, the SGT did not present a useful result. It was the work process that was educational. Knowledge about the system could have been caught more efficiently in another method. In contrast, the result from the heuristic evaluation was easy to analyze and work with.

The outcome from the heuristic evaluation and the SGT were, however, not enough to serve the users expectations of the system. The user test revealed that much was missed in these evaluations that were important to the users. Since the result from the user test had high influence in the second design proposition it was unlike the first.

The importance of letting the users be part of the design process should never be underestimated. This was certainly revealed in this study since the design proposition after the SGT and heuristic evaluation was not approved to a great extent by the users. The parts that the users were most satisfied with in the first design suggestion was the graphics of the interface, however the functionality was not what they expected. Therefore the three evaluation methods complemented each other.

It is hard to develop a system image that represents the designer's conceptual model. There are always obstacles, for example implementation or graphical limitations that have to be considered. In this project several limitations restricted the work. For instance the static VIS button in the first design proposition reduced the number of button spaces to other functions.

For the designer the main purpose of evaluations is to better understand the system and from the knowledge gained develop new propositions. Therefore the iterative design process is efficient in finding usability problems. Even though the last design proposition always is the "optimal" for the moment, it is useful to exploit it and further evaluate to find more design improvements in the next session. There are always improvements to make.

5.1 Limitations

A limitation in this project was a common problem - not enough time! It is hard to plan a project where much is unknown and sessions that are expected to last one week may suddenly last two and so on. The programming part of the project was the most time consuming; it lasted six weeks or 1/3 of the project time. The implementation session was however hard to cut down on since a half-made prototype is not useful in a user test.

5.1.1 SGT

The greatest limitation with the SGT was that the input to the task was too weak in the sense of being adapted to the task. This made the result hard to interpret and therefore not very useful.

5.1.2 Heuristic Evaluation

The result from the heuristic evaluation was very useful when making the first design proposition. However the flaws of the method emerged in the user test. A heuristic evaluation should never be done without a user test to complement it. The result could have been better with more experts than just one doing the evaluation, but the knowledge the users have about the system is invaluable and could probably not be found even if hundreds of expert evaluators were used.

5.1.3 User Test

A lack in the user test was that not every part of the GUI was implemented in the prototype. This made the users a bit confused whether certain functions were to be considered or not. Another limitation with the user test was that the result of the questionnaire was quite hard to interpret. A defect in the materials to the written user test was that the questionnaire was too long and therefore the comments became fewer and fewer towards the end of the inquiry.

5.2 Future Work

The prototype does not have all functions implemented and the format of the prototype is not tested in the touch screen hardware. Therefore the prototype has to be tested and expanded to fit in the vehicle.

Since the design work has been concentrated to the gunner's work station. A part of future work is to consider the commander's work station. The graphics will probably be similar, but certain functions will be hidden, for instance the fire settings should be locked in the commander's work station.

The iterative design process has not stopped while finishing this master thesis. The second design proposition could also be tested with the users and then a new proposition can be designed and so on.

Another proposition for future work is to improve the way to use the template for heuristic evaluations, which is provided in this report, in the design process at Hägglunds.

Chapter 6

Acknowledgements

I would like to thank a number of persons for their direct or indirect help in doing this master thesis. In no particular order:

Daniel Granholm

My supervisor and mentor at Hägglunds. Thank you for being so helpful. Your comments were very useful.

Lena Palmquist

My supervisor at Umeå University. Excellent help with this report.

Stefan Westman

You were a great help in explaining the system.

The test participants

Your help was invaluable.

Andreas Anundi and Anders Holm

It was fun and educational working with you guys. Good luck in the future.

Family and Friends

Thank you for all your support. Especially I would like to thank Andreas for great support and for understanding the priority of this thesis.

References

[1] J. Annett. Hierarchical Task Analysis. In D. Diaper and N. Stanton, editors, *The Handbook of Task Analysis for Human-Computer Interaction*, pages 67–82. Lawrence Erlbaum Associates, Inc., Publishers, New Jersey, 2004.

[2] J. Annett and K. D. Duncan. Task analysis and training design. *Occupational Psychology*, 41:211–221, 1967.

[3] A. Anundi and A. Holm. Creating and evaluating a framework for HCI-aspects in design in use case driven systems development. Master Thesis. Technical Report UMNAD-602, Dept. of Computing Science, Umeå University, Umeå, Sweden, 2005.

[4] G. Bailey. Iterative methodology and designer training in human-computer interface design. In *Proceedings of the Conference on Human Factors in Computer Systems*, pages 198–205, New York, 1993. ACM.

[5] RW. Bailey, RW. Allan, and P. Raiello. Usability testing vs. heuristic evaluation: A head-to-head comparison. In *Proceedings of the Human Factors Society 36th Annual Meeting*, pages 409–413, 1992.

[6] S. Bly. Interview with Sara Bly. In J. Preece, Y. Rogers, and H. Sharp, editors, *Interaction Design - beyond human-computer interaction*, pages 387–388. John Wiley and Sons, New Jersey, NJ, 2002.

[7] H. Desurvire, J. Kondziela, and M. Atwood. What is gained and lost when using evaluation methods other than empirical testing. In *Proceedings of HCI'92*, pages 1–16, York, England, 1992.

[8] J. S. Dumas and J. C. Redish. *A Practical Guide to Usability Testing.* (Revised edition). Intellect Books, Exeter, UK, 1999.

[9] J. Preece (ed), Y. Rogers, H. Sharp, D. Benyon, S. Holland, and T. Carey. *Human-Computer Interaction.* Addison-Wesley, Harlow, England, 1994.

[10] R. Jeffries and H. Desurvire. Usability testing vs. heuristic evaluation: Was there a contest? *SIGCHI Bulletin*, 24(4):39–41, 1992.

[11] R. Jeffries, J. R. Miller, C. Wharton, and K. Uyeda. User interface evaluation in the real world: A comparison of four techniques. In *Proceedings of ACM CHI'91*, pages 119–124, 1991.

[12] D. Liddle. Design of the conceptual model. In T. Winograd, editor, *Bringing Design to Software*, pages 17–31. Reading, MA: Addison-Wesley, 1996.

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

[14] J. Nielsen. Finding usability problems through heuristic evaluation. In *Proceedings of ACM CHI'92*, pages 373–380, 1992.

[15] J. Nielsen. Enhancing the explanatory power of usability heuristics. In *Proceedings of ACM CHI'94*, pages 152–158, 1994.

[16] J. Nielsen. Technology transfer of heuristic evaluation and usability inspection. In *Proceedings of the IFIP Interact'95 International Conference on Human-Computer Interaction*, 1995.

[17] J. Nielsen and R. L Mack (eds). *Usability Inspection Methods*. John Wiley and Sons, New Jersey, NJ, 1994.

[18] J. Nielsen and R. Molich. Heuristic evaluation of user interfaces. In *Proceedings of the ACM CHI'90*, pages 249–256, 1990.

[19] D. A. Norman. *The Design of Everyday Things*. The MIT Press, third printing 2000, London, England, 1988.

[20] D. A. Norman. Affordances, Conventions and Design. *Interactions*, 6(3):38–43, 1999.

[21] T. C. Ormerod and A. Shepard. Using task analysis for information requirements specification: The sub-goal template (SGT) method. In D. Diaper and N. Stanton, editors, *The Handbook of Task Analysis for Human-Computer Interaction*, pages 347–365. Lawrence Erlbaum Associates, Inc., Publishers, New Jersey, 2004.

[22] S. J. Payne. Users' mental models: The very ideas. In J. M. Carroll, editor, *HCI Models, Theories, and Frameworks toward a multidisciplinary science*, pages 135–156. Morgan Kaufmann Publishers, San Fransisco, USA, 2003.

[23] J. Preece, Y. Rogers, and H. Sharp (eds). *Interaction Design - beyond human-computer interaction*. John Wiley and Sons, New Jersey, NJ, 2002.

[24] M. Rettig. Prototyping for tiny fingers. *Communications of the ACM*, 37(4):21–27, 1994.

[25] D. Schön. *The Reflective Practitioner: How Professionals Think in Action*. Basic Books, New York, 1983.

[26] The "Usability First" website search for design-prototype-test cycle. http://www.usabilityfirst.com/glossary, last visited 2005-11-07.

[27] The "Usability First" website search for iterative design. http://www.usabilityfirst.com/glossary, last visited 2005-11-07.

[28] J. Whiteside, J. Bennett, and K. Holteblatt. Usability engineering: our experience and evolution. In M. Helander, editor, *Handbook of Human-Computer Interaction*, pages 791–817. Elsevier Science Publishers, Amsterdam, 1988.

Appendix A

A Template for Heuristic Evaluation

A.1 Graphics

A.1.1 How to Use Color

Color is useful for indicating different kind of information, i.e. cueing. Therefore color should not be used excessively because then the effect of the color-coding will decrease.

1. Is color used as a form of coding?
    1 How is it used as coding?
2. Is the color (used in the user interface) excessive and garish?

A.1.2 Simplicity

It is important not to have too many design elements in the interface, since it will be messy and hard to get a general view of the system. Sometimes it is better to remove design elements such as icons, buttons, boxes, lines, graphics, shading, and text, to make it cleaner, crisper, and make it easier to navigate the user interface. Furthermore, if too much graphics is used the download time will increase. Animations should only be used if necessary to have the user's attention. To make the user interface more usable, the system should not provide information that is irrelevant or rarely needed. However, some "unnecessary" graphics are not appropriate to be removed since it can make the user interface more aesthetic attractive and therefore more easy to navigate and read. Therefore the most important thing is to find a balance between aesthetic appeal and the right kind of information amount on the user interface.

1. Are there lots of graphics?
    1 Do the graphics add to the user interface?
    2 How long does it take to load each page?
2. Are there any flashing animations?

    1 Are the animations used to attract the user's attention or are they misused?

3. Are there any complex introduction sequences?

    1 Can the introduction sequence be short-circuited?

4. Is any unnecessary and irrelevant information provided?

A.1.3 Consistency

The design should be consistent in how to use fonts, numbering, terminology, etc. This is important for usability and for aesthetically pleasing designs. The user should not have to wonder whether different words, situations, or actions mean the same thing.

1. Are the same buttons, fonts, numbers, menu styles, etc. used across the user interface?

2. Are the components used in the same way?

3. Are menus named consistently?

A.1.4 Constraints

The concept of constraints in interaction design is to force the user to do or not to do an action. When discussing graphics in an interactive interface the most interesting constraint is the cultural constraint which is in learned conventions, e.g. the color red means "warning" and the smiley :-) means "happy".

1. Can the colors/symbols etc. be misinterpreted?

A.2 Structure

A.2.1 Simplicity

Reading long sentences, paragraphs, and documents is difficult on screen, so break material into discrete, meaningful chunks to give the user interface structure.

1. Is the page layout structured meaningfully?

2. Is there too much text on each page?

A.2.2 Avoid Dead-Ends

Avoid orphan pages i.e. pages that are not connected to the home page, because they lead users into dead ends.

1. Are there any orphan pages?

    1 Where do they go?

A.2.3 Avoid Narrow Menus

Avoid narrow, deep, hierarchical menus that force users to burrow deep into the menu structure.

1. Are there any narrow, deep, hierarchical menus?

A.2.4 Consistency

Consistency within the user interface (e.g. navigation, menu names, etc.) is important for usability and for aesthetically pleasing designs. Also avoid making users wonder whether different words, situations, or actions mean the same thing. Besides this, provide consistent look and feel for navigation and information design.

1. Are the same buttons, fonts, numbers, menu styles, etc. used in the same way?
2. Are there ways of performing similar actions consistently?
3. Are menus used and positioned consistently?

A.3 Cognitive Help for the User

A.3.1 Safety

The system should be safe to use and protect the user from dangerous conditions (e.g. X-ray machines) and undesirable situations (e.g. accidentally erase all mails in the inbox).

1. Does the system prevent users from making serious errors and, if they do make an error, does it permit them to recover at all?

A.3.2 Learnability/Effectiveness

An interactive system should be effective to use in the aspect of how good the system is at what it is doing. This will make the user interface more easy to learn.

1. How easy is it and how long does it take to get started using a system to perform core tasks?
2. How easy is it and how long does it take to learn the range of operations to perform a wider set of tasks?
3. Is the system capable of allowing people to learn well, carry out their work efficiently, access the information they need, do the tasks they want to do etc.

A.3.3 Memorability/Efficiency

The system should be efficient to use in the way the system supports users to manage their tasks. This helps the user to remember how to use the system once leaned.

1. What kinds of interface support have been provided to help users remember how to carry out tasks, especially for systems and operations that are used infrequently?
2. Once users have learned how to use a system to carry out their tasks, can they sustain a high level of productivity?

A.3.4 Error Messages

Help users recognize, diagnose, and recover from errors by using plain language to describe the nature of the problem and suggest a way of solving it.

1. Are error messages helpful?

2. Do the error messages use plain language to describe the nature of the problem and suggest a way of solving it?

A.3.5 Visibility and Feedback

By supplying information that can be easily searched and provide help in a set of concrete steps that can easily be followed the user can be helped to navigate the user interface. To help the user to know what to do next the functions should be visible. The user should always be informed about what is going on. The system should inspire recognition rather than recall to help the user.

Feedback is sending back information about what actions have done and what they have managed to do, while the user is still involved in the activity. Examples of types of feedback in interaction design are audio, tactile, verbal, visual, and combinations of these. Since feedback is related to the concept of visibility, feedback can help provide the necessary visibility for the user interaction.

1. Is help information provided that can be easily searched and easily followed?

2. Are users kept informed about what is going on?

3. Are objects, actions and options always visible?

4. Is appropriate feedback provided within reasonable time about a user's action?

A.3.6 Mapping

Mapping concerns the relationship between controls and their effects in the world. The controls should be placed in an established way that is accepted by the user. In Figure A.1 four different placements of the controls are shown. Figure A.1d is preferred before A.1c.

1. Are the controls in the user interface presented with a good mapping?

A.3.7 Language

The system should speak the user's language, using words, phrases and concepts familiar to the user, rather than system-oriented terms.

1. Is the language used in the interface simple enough?

2. Are words, phrases and concepts used familiar to the user?

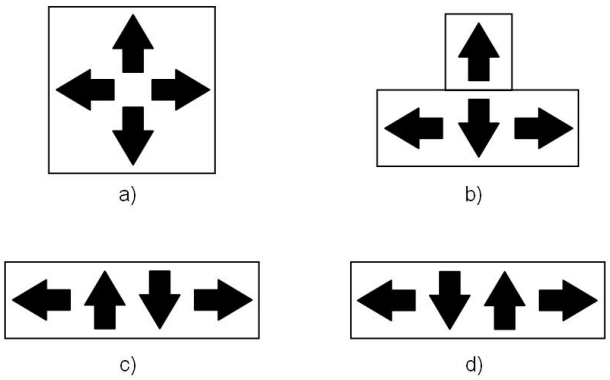

Figure A.1: Example of mapping. Four different placements of controls.

A.4 Navigation Support

A.4.1 Guidance

The system should provide navigation support, such as a site map that is always present.

1. Is there any guidance, e.g. maps, navigation bar, menus, to help users find their way around the site?

A.4.2 Exploration

To give the user control and have freedom the system should provide ways of allowing users to easily escape from places they unexpectedly find themselves in, by using clearly marked “emergency exits”.

1. Are there ways of allowing users to easily escape from places they unexpectedly find themselves in?

A.4.3 Flexibility

Provide accelerators (i.e., shortcuts) that are invisible to novice users, but allow more experienced users to carry out tasks more quickly.

1. Have accelerators been provided that allow more experienced users to carry out tasks more quickly?

A.4.4 Error Prevention

Error prevention is very important. If possible, the developers of the system should prevent errors from occurring in the first place.

1. Is it easy to make errors?

    1 If so where and why?

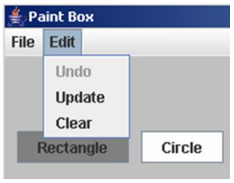

Figure A.2: Example of a logical constraint. The Undo action is inactivated because it is not possible/allowed to perform the Undo-action.

A.4.5 Constraints

The concept of constraints in interaction design is to force the user to do or not to do an action.

When discussing navigating an interactive interface the most interesting constraint is the logical constraint. The concept of logical constraint is the way people understand how the world works and to hide actions that are not allowed to execute. An example of this is disabling menu options in the menu bar when they are not appropriate to execute, see Figure A.2.

1. Are the users "locked" to do actions that are not available?

A.5 System Knowledge

A.5.1 Utility

A system should have good utility to such an extent that the system provides the right kind of functionality so that users can do what they need or want to do.

1. Does the system provide an appropriate set of functions that enables users to carry out all their tasks in the way they want to do them?