Redesigning a graphical user interface for usage in challenging environments with a user-centered design process

ALEXANDER PERSSON
Redesigning a graphical user interface for usage in challenging environments with a user-centered design process

Master in Computer Science
Date: September 9, 2019
Supervisor at KTH: Jonas Moll
Supervisor at ÅF: Mats Elfving
Examiner: Joakim Gustafsson
School of Electrical Engineering and Computer Science
Host company: ÅF Digital Solutions AB

Swedish title: Design av gränssnitt anpassat för användning i utmanande förhållanden.
Abstract

Different possible interactions with computers is an ever-evolving topic and the usage of computers are more ubiquitous than ever. To design with the user in mind is not an easy task with regular use cases and interactions at hand. Designing for users in a military context can be even more difficult as the working environment of said users is demanding. This thesis sets out to investigate how a redesign of an existing GUI can reduce the impact of the contextual challenging environment of operating a software in terrain vehicles and in outdoor weather. For the redesign of the GUI a user-centered design process was performed. The process was initiated by using the method of contextual interviews and affinity diagram for data gathering and analysis, which gave a deeper understanding of the user’s issues and needs. After defining the different key elements for the redesign, a prototype was developed. The first prototype was evaluated by experienced users of the software out of military context. With the feedback from the users another developed version of the software was created and evaluated by current users of the software with an interview in military context. The evaluation showed that the users believed that the redesign of the GUI would help mitigate problems caused by the challenging context the software is used in, as well as improve quality of work.
Sammanfattning

Olika möjligheter för interaktion med datorer är ett område i ständig utveckling och användandet av datorer idag är mer omfattande än någonsin. Även för vanliga användningsområden och interaktioner så är det svårt att designa något där användarens behov står i centrum. Att designa något för en militär kontext kan göra processen ännu svårare då arbetsmiljön för användarna är mycket påfrestande. Denna studie undersöker hur en ny design av ett grafiskt användargränssnitt kan reducera de negativa effekterna som uppstår i den kontextuella miljön. Den miljö som datorprogrammet används i är i terrängfordon i rörelse och i alla väderförhållanden utomhus i fält. För att skapa en ny design av gränssnittet utfördes en användarcentrerad designprocess. Processen initierades med att utföra kontextuella intervjuer och skapa ett affinitetsdiagram för att sammanställa och analysera data, med detta erhölls en djupare insikt av användarbehoven och krav för designen. Efter att de mest betydande huvuddragen för designen var fastställda utvecklades en prototyp av designen. Första prototypen utvärderades av användare med erfarenhet av mjukvaran utanför den militära kontexten. Av responsen från dessa användare utvecklades en ny version av designen som sedan utvärderades under en intervju med nuvarande användare av mjukvaran i militär kontext. Utvärderingen visade att användarna trodde att den nya designen av användargränssnittet skulle hjälpa att minska problemen som uppstod i deras arbetsmiljö och kontext samt förbättra deras arbetskvalitet.

Keywords

User-centered design, Graphical user interfaces, Military communication systems
# Contents

1 Introduction .............................................................................................................1  
1.1 PC-Dart ...........................................................................................................2  
1.2 Purpose .............................................................................................................4  
1.3 Problem statement .............................................................................................4  
1.4 Delimitations .....................................................................................................4  
1.5 Sustainability and Ethics ..................................................................................5  
2 Related work ..........................................................................................................6  
2.1 User-centered design process ................................................................ ..........6  
  2.1.1 Contextual interviewing .............................................................................7  
  2.1.2 Data interpretation and the affinity diagram ...........................................8  
  2.1.3 Prototyping .................................................................................................9  
  2.1.4 Design evaluation .......................................................................................9  
2.2 Contextual HCI ..................................................................................................11  
  2.2.1 Military HCI ...............................................................................................11  
  2.2.3 Responsive and adaptive user interfaces ................................................12  
  2.2.4 Designing for portable interaction ............................................................13  
  2.2.5 Interactions in challenging environments ................................................13  
3 Method ....................................................................................................................15  
  3.1 Contextual interviews in vehicles ..................................................................15  
  3.2 Affinity diagram ...............................................................................................17  
  3.3 Prototyping and implementation of the design ..............................................17  
    3.3.1 Prototyping ...............................................................................................17  
    3.3.2 Development and implementation of the design .....................................18  
  3.4 Evaluation of preliminary developed design ................................................18  
  3.5 Evaluation of the final design ..........................................................................19  
4 Results ....................................................................................................................20  
  4.1 Analysis of contextual interview and affinity diagram ..................................20  
    4.1.1 The user’s perception of the software ....................................................20  
    4.1.2 The user’s context ...................................................................................21  
    4.1.3 GUI and interaction with the software ...................................................22  
    4.1.4 Hardware and input devices ....................................................................23  
    4.1.5 The software’s functionality .....................................................................23  
  4.2 Identified problems/Proposed solutions with the GUI ..................................24
Chapter 1

Introduction

Graphical user interfaces (GUI) come in all shapes and forms from the first commercial introduction by Xerox Parc with the Xerox Alto personal computer [1] to the groundbreaking portable user interface of the Apple iPhone. As computers and hardware developed over time into a more diverse and broad area so did the possible ways of interaction with them. This poses a challenge for the designer of the GUI with a heterogeneity issue [2], meaning that the designer has to take all the different possibilities for platforms, inputs and use cases into account when implementing a design. The consumer market for computers moves quickly with cheap options for upgrading systems into newer ones. In large scale operations such as the military or corporations, replacing systems and hardware is often complicated and costly in an ever-changing environment [3]. Sweden has a long-lasting trend in decreased military spending as a share of GDP [4] and with this comes a tight budget. In some situations, the only choice is to keep old hardware and build systems around it since it is still functional and much less costly than replacing a complete system. This leads to software designed for certain platforms in which it works great. But with the expanded use case of the software including portable devices, the functionality starts to lack with the presentation of the GUI and the user's capability to efficiently navigate through the software's functionality with its input devices. Screen elements and buttons designed to be navigated with a mouse pointer turn out difficult to click accurately on a touch screen. Different screen sizes and resolutions as well as the user's distance from the screen affects the user's capability to read text on screen. This means that the text on a screen can be of sufficient size and readable on one device but completely indecipherable on another. This leads to poor user experience and usability of the software which is a problem, especially so when the software is necessary for the users to do their job and when they have no other alternative software to use. This thesis seeks to explore how to redesign and to improve the user experience of a previously non-adaptive software for portable platforms under the circumstances of a challenging environment.
1.1 PC-Dart

The software this thesis investigates and redesign is the program PC-Dart. PC-Dart is a communication software designed for and used by the Swedish Armed Forces [5]. Its primary purpose is to send and receive short text messages over radio and IP-interfaces (see Figure 1.1).

![Figure 1.1: PC-Dart, standard window setup, Swedish GUI](image)

The messages, also called DART-messages (Data Report Terminal- messages) are of different predetermined formats including fields for information input such as recipient, sender, text etc. Each format corresponds to different information and actions to be obtained by the receiver of the message, such as permission to fire for artillery units or positional reports (see Figure 1.2).
The software was created as a PC-program to upgrade/replace the old system DART 380 which is a computer terminal with the specific purpose to send DART-messages. The PC-Dart software is connected to external radio hardware such as Radio 180 (military radio, used by the Swedish armed forces) through the computer's serial/ethernet/audio ports in order to send and receive the DART-messages. The PC-Dart software is built for and run on the Windows operating system. The program is currently used in several different versions depending on where it is used, but the software is fully backward-compatible to its previous versions. PC-Dart is currently used on a wide range of different hardware platforms, including rugged portable tablet computers designed to be mounted in vehicles or carried in field, but also on regular desktop/laptop computers for stationary usage. The general aspects of the GUI of PC-Dart is based on the legacy hardware DART 380 in style and function, this was done in order to keep overlap between the two for easier transition between the systems. PC-Dart’s GUI is non adaptive and non-responsive and is best suited for desktop usage with a sufficiently large monitor as output device, keyboard and mouse as input devices.
1.2 Purpose

The purpose of this thesis is to apply user-centered design methods to redesign the GUI of the military communication application PC-Dart. The redesign of the application shall be adapted to usage on portable platforms in the demanding conditions and environments such as in moving vehicles in terrain or outdoor portable usage. Furthermore, this study seeks to provide general guidelines for GUI design for usage in demanding environments.

1.3 Problem statement

When using an application the GUI of the program will convey the usage and functions of the software, the design of it can either make or break the software as a whole. A GUI designed for a desktop platform can differ much on a mobile platform and vice versa. A poorly adapted solution for the situation can make the user experience and efficiency of the user suffer. The same goes with the context of usage for the software, different situations and conditions brings different effective means of interaction. This thesis seeks out to answer how to design for some of these circumstances, and the research question is as follows:

*In which ways can a redesign of an existing GUI, based on user-centered design methods, reduce the impact of challenging environments on usability when using an application on portable platforms?*

1.4 Delimitations

As the thesis work is limited to a timespan of 20 weeks, some delimitations of the project were needed to answer the research question. The redesign of the GUI is limited to the PC-Dart software and aimed towards a tablet/portable device. This implies that the research is aimed towards a military context which will affect the results in terms of development and future guidelines on designing for challenging environments. As the PC-Dart software is used extensively within the military there are many different user groups in terms of where and how the software is used. This research will focus on the user groups performing a subset of tasks in the challenging environments of outdoor portable usage and in moving military vehicles.

Finding users with real experience and with time available to investigate the design on is difficult, involvement of users in contextual environment was limited to two occasions. The first occasion was for interviewing for requirements elicitation before development and the second one was for post development design evaluation. Out of context data from users were gathered continuously throughout the development process.
1.5 Sustainability and Ethics

The Swedish Armed Forces is a tax funded authority with the task to defend Sweden and secure its borders and peace as well as partake in international peacekeeping efforts supporting security around the world. Systems for a military context such as the application PC-Dart for the Swedish Armed Forces serves a great necessity for successful communication between different operational parts of the organization. Working in challenging environments and context as often the case is for military operations, efforts should be made to provide optimal conditions for the forces to focus on their work, this includes the system and gear available. As the organization is on a budget and funded by Swedish taxpayers, the spending for the organization is carefully carried out to provide the best core service possible. This creates the need for sustainable cost-efficient solutions that can be maintained for a very long time. PC-Dart being a great example of the fact with over 20 years in use and sustained capability. This project seeks out to explore the possibilities to improve the work when using the PC-Dart software in its context, and more generally improve GUIs used in challenging environments. This does not only affect the PC-Dart software but with potential to reuse key findings for similar systems. This could improve the communication capabilities which leads to a more sustainable work environment for the users and improve the organization and its peacekeeping efforts globally. From a developer point of view in working with systems for a military context there are some ethical perspectives present to take in mind. As the systems developed often is based in some secrecy due to the nature of the context, the developer needs to be comfortable with this fact. Some parts in their daily work might be considered secret and violation of the secrecy could lead to a sentence in a court of law.
Chapter 2

Related work

The following chapter describes some of the methods that can be used for determining how to design a GUI as well as general research within the area of GUI design related to the topic of design for contextual intricacies. The first section “2.1 User-centered design process” describes how a user-centered design process is carried out and what phases the designer goes through in the process. Some UCD-methods used in these phases are described in detail and how they are used, why they are used and when they are used in order to create a successful design. The second section “2.2 Contextual HCI” will go into detail and explore research done in areas related to general contextual HCI as well as take a closer look into the specific context in focus of this thesis, military applications and small portable platforms.

2.1 User-centered design process

To be able to establish a foundation for the design of a GUI different design methods can be applied to the process. One prominent group of methods belongs to the user-centered design process [6]. In this design process the focus lies on the users of said design, the goal is to create highly usable and accessible products by involving the user throughout the process. The key for this process to work is iteration, to take the user's thoughts and opinions into consideration, reevaluating the steps of the process based on the user’s feedback and present the results for the users to evaluate again. A general user-centered design process consists of four separate phases [6] (see Figure 2.1). The goal of the first phase is to try to understand in which context the users use the system. This then leads to the second phase in which the designer identifies and specifies the user’s requirements. This initiates the third phase and the design work of the process in which the development of the solution starts, based on the previous established requirements from phase two. The fourth and final phase is the evaluation phase where the design outcome is evaluated against and with the users to determine how well the design fits the user’s context and satisfies the user's needs. Depending on the results of the evaluation, the phases of the process are repeated until a satisfactory result is achieved.
The following sections will describe in more detail some of the methods that can be used in a user-centered design process for each phase, from the user's context to a completed design.

### 2.1.1 Contextual interviewing

The basic use-cases of a device or software can often be inferred by inspecting and looking at it, going through the core functionality and components. Traditional interviews often give great amounts of data, but the quality of said data can differ depending on how the interview is done. A mistake made in interviewing is to make the interviewee in question reminiscent or remember past usage of a design or product which often leads to the user making up a rational story of what they think the interviewer would want to hear instead of how they actually used it [7]. Another common pitfall is trying to make the users imagine the future of said product as it is difficult to envision something that does not yet exist based on a description alone, there is no way of telling how the user will perceive a design ahead of its existence [7]. A focus for interviewing users should lie on how to get a deeper knowledge on how the users actually use an application and how they perceive it in its context, which fits right in with the first phase of a user-centered design process. The method of contextual interviewing was established to do just this.

The contextual interview or inquiry seeks to capture the world of the user, exploring this deeper contextual knowledge. The method is described in practice as approach the users in their context, to watch and observe how they are conducting their work/ using a system and to talk to the users about it. This will in turn will give a greater insight of your user [8]. A contextual interview is carried out by coming to the user in their context, in which the interviewer, unlike the traditional relationship of interviewer/interviewee, tries to establish a master/apprentice model [8]. This model promotes an inclination of inquiry for the apprentice and an attitude of sharing for the master. The master teaches by showing its craft and telling what is going on and the apprentice asks questions to understand and learn about the actions in context. During the inquiry notes are

![Figure 2.1: The user-centered design process](image)
to be taken as well as supplementary recordings and photos to get a vast amount of data to break down post interview.

### 2.1.2 Data interpretation and the affinity diagram

After the first phase of the user-centered design process the user’s context has been well established. Beyond the experience of the context of the user that the designer has taken part of, the interviews give unstructured data. This unstructured data is to be analyzed and structured in a manner that enables the designer to identify and specify the user’s requirements. An example of a method used for this second phase of the UCD-process is constructing an affinity diagram [8]. Tightly coupled with the contextual interview the affinity diagram is done by grouping items or notes of data in a bottom up perspective, meaning starting with individual items and finding structures and similarities in which you can categorize them. These categories then make up groups for other categories creating a tree like structure of the data gathered (see Figure 2.2). This diagram will then be able to communicate the user’s perspective and requirements in a systematic way and can be designed in the same manner. Using the competence of a cross-functional team with expertise from areas such as product management, marketing, user experience designers etc. when performing this method is advised as each individuals unique skills and insights can help getting closer to a design that is right for the users [8].

![Affinity Diagram](image.png)

*Figure 2.2: Affinity diagram structure*
2.1.3 Prototyping

With clear categorized and analyzed data with user insights and requirements, a solid foundation is in place to initiate the third phase of the UCD-process. A method to start this phase with is by constructing a prototype design based on said insights. Prototyping a product or a design is a way of shortening the feedback cycle of the fourth phase of the UCD-process, compared to going to development work and code writing directly after the second phase of the process. This enables for quicker iterations of the design involving the users earlier in the process. Furthermore, applying the method of participatory design and involve users in the prototyping phase can be effectively used to shorten the process even further by integrating changes in the prototype [9]. This can help to avoid pitfalls that might not be apparent to the designers until the users point them out. This can lead to satisfying the user requirements quicker [6].

There are often several different levels of prototypes created to establish a design [9]. Prototypes can be rapidly constructed low-fi prototypes on paper illustrating a design all while enabling users to give instant feedback and even enabling the user to help the designer change the design in an instant. This is often utilized in an early stage of design to be able to visualize what might work and what will not. But a prototype can also be more intricate with a wireframe, which is a method in which the full interface is laid out with key elements but without the content of the product or the stylization set by the designer. Even more work can be put into a wireframe to turn it into an advanced hi-fi prototype in the sense that it looks like and works like a final product with the exception of the actual code and logic not being implemented as in the case of software development. This gives the designer an opportunity to evaluate and test the design on users, as the users would approach the design in the same way as they would if it were a developed prototype, further verifying the design to be ready to be developed.

2.1.4 Design evaluation

Beyond the third phase of development of a design, or as described above with the iterative process of prototyping, the fourth phase of evaluation of the design is initiated. There are various methods that can be used to evaluate designs within the UCD-process, different methods can capture flaws or functions in the design in unique ways leading the designer towards a satisfactory design.

One method of design evaluation is usability testing. This method can help to verify the design choices and expose flaws in the design. Usability tests are typically done with users that are representative of the target group as a whole. The users perform tasks as they typically would and data from this is collected with a balance of observation, quantitative and self-reporting data [9]. A framework that can be used for usability quality testing consists of the five components memorability, efficiency, errors, learnability, and satisfaction (MEELS) [10]:

- **Memorability**, with infrequent usage of the design, how easy can the users restore their proficiency with the design?
- **Efficiency**, once the users have learned the design, how quickly can they perform the tasks at hand?
- **Errors**, how many errors do users make, how critical are these errors for the design and how do users recover from these errors?
- **Learnability**, how long does it take for the user to learn and use the design once it has been presented to them?
- **Satisfaction**, how pleasant do the users perceive the design?

Beyond usability there are several other aspects to take in mind when evaluating the quality of a design, one being the utility of the design. The utility of a design can be described as the functionality, what does the user need the product to do? A method for evaluating the utility of a design is utility inspection method (UIM) [11]. The method consists of seven heuristics with the idea to summarize the data and serve as a checklist of the evaluation in terms of utility. The heuristics are the following:

- **Global platform**, availability on different devices and communications with other technology and systems.
- **Infrastructure**, maximized system performance and information to the user, making the system supporting for the user’s productivity.
- **Robustness**, stable basic functionality with a fast learning curve allowing users to learn the system fast and how to use it.
- **Rich content**, the content of the design should be contained to important and meaningful things that let the users express their desires in an efficient way.
- **Customization**, the system should allow users to customize the design in a way that is tailored to their context or for specific usage, the system should adapt to the user’s context as well.
- **Symbolizing**, the users should be able to influence the system look/behavior to reflect their personal identity on a social level.
- **Companionship**, the system should contain clear communication between users as well as possibilities for the users to influence the design of the software by comments of improvements to the developers.

Each one of the seven heuristics are not always applicable to every design and the possibility of more heuristics for some designs are plausible. Some of the heuristics cover both utility as well as usability as these are closely related. Usage of UIM can help to bring out more important context related problems of a design compared to using the heuristic evaluation method, which is another popular method for usability testing [12]. This goes at the expense of identifying less problems about the interface than heuristic evaluation [11].

## 2.2 Contextual HCI

As the spectrum of different HCI devices expand over time so does the use case and context in which they are used. The approach of creating something accessible and usable for the user on a desktop might not fit a user in another context. A context aware interface gathers information on its context of usage whether it is by external factors such as connected appliances [13] to the system
or internal factors by device recognition [14]. An example of this is found in a study of an assistive technology system in which a context aware GUI was successfully created by recognizing remote appliances within range of the device which then prompted and notified the user through the GUI [13]. The PC-Dart software described in section “1.1 PC-Dart” is context aware in the sense that its connectivity to different radio modules and external interfaces determines the functionality available in the software, but not context aware in terms of the GUI.

Another aspect is when the context demands the user to interact with a system in a particular way. A study on HCI in fieldwork environments brings up this aspect in which four characteristics of a fieldworker using a portable device are described as the following [15]:

- **Dynamic user configuration**, it is unlikely that a user can set up a traditional working space during field work. The fieldworker should be able to use the system whether standing, crawling or walking etc.
- **Limited attention capacity**, fieldworkers often have their surroundings to pay attention to when observing to collect data which limits their capability to interact with a system for prolonged time.
- **High-speed interaction**, a fieldworker must be able to quickly record data as it comes in bursts - the interaction with the system needs to be fast and accurate.
- **Context dependency**, the fieldworker’s activities depend on their context, the data gathered is generally self-describing in terms of context and is often analyzed in its context.

Each one of these characteristics can be designed for in order to maximize the user’s proficiency when working and collecting data in the field. A core principle of these characteristics is that the context really determines the use case in the situation and how the design should be done. The area of fieldwork can be broadened to categories of work with similar characteristics.

### 2.2.1 Military HCI

In a military context such as usage of the PC-dart software described in section “1.1 PC-Dart” many of the above mentioned attributes for fieldwork from section “2.2 Contextual HCI” can be applied in the same way on a system, setting specific requirements. In addition, a military context features additional underlying challenges and hurdles to overcome in order to make a design that works great for the users. In the process of creating military systems the user’s perspective often gets neglected in the vast requirements of hardware components and software specification [16]. It takes copious amounts of work to make good requirement specifications even without the users being directly involved. Furthermore, to get the user involved in the design and testing at earlier stages of development is often a difficult factor with military applications as there often is significant distance between the end users and the development project [17]. If the user is involved from the start the challenge lies in capturing the full context for the work since it often involves work in field during prolonged periods of time.
As for portable military systems there is some extra context specific considerations to take in mind when designing. One of the most prominent characteristics you have to take in mind is the limited screen size leading to limited information that is visible to the user, thereby making it important to carefully analyze what functionality that must be present [18]. Another perspective is that the input of the device often is limited to touch-screens, miniature keyboards, or pen-based handwriting systems, with each one of them creating different problems in different situations [18]. Other factors related to usability of military devices is that they often have to be more rugged and durable in order to withstand the challenging environments in which the devices are used, which often causes long development times and at an extra cost for the purchaser. Additionally, these devices tend to be heavier than their commercial counterparts reducing the mobility of the user that is already weighed down by other military appliances and gear [18].

2.2.3 Responsive and adaptive user interfaces

An application with a broad use case and availability in terms of platforms leads to different use cases between the users. Responsive design is an often used way to adapt the GUI presented to the user to fit the format of the platform, e.g. the same information presented on a desktop PC gets compressed to fit the screen of a mobile. The general idea with responsive design is that the GUI should look the same, feel the same and work the same regardless of screen size, orientation or target platform. While there often is a differentiation in modern software between a desktop layout of a GUI and a mobile version, generally the same information is conveyed to users of the software.

Yet in some cases the usage of software is not always static between different platforms, a solution to solve this differentiation is to have an adaptive interface across different platforms and users for the software. A Role-Based UI Simplification (RBUIS) [19], a structure for increasing usability by providing users the optimal GUI based on just their specific context, can be utilized for this. A minimal number of features required for the user to complete its job is determined, and an optimal layout is to be drawn from it. By taking the context into account by changing the GUI to fit the situation and user, the most relevant information can be enhanced, and the irrelevant parts can be safely obscured in the GUI. An adaptive interface utilizing RBUIS could make the user experience more efficient by only presenting the most relevant information to the user reducing the cognitive load of filtering the needed information for the user [19].

Adaptive GUIs can improve the user’s efficiency in their work but can also introduce an uncertainty in that the recurrent adaptations reduce the consistency of the GUI, which reduces learning ability for the users [20]. Users also generally perceive an automatically adapted GUI as less aesthetically pleasing compared to a designer made GUI [20]. Although the application and usage of Adaptive GUIs improve the user’s efficiency it needs to be under controlled circumstances in according to be coherent with the rest of the design and to avoid over adaptation.
2.2.4 Designing for portable interaction

As previously mentioned in section “2.2.1 Military HCI” a key aspect of designing for portable devices is the limitation of screen size. A prominent problem occurring on such devices is the “fat finger” problem [21]. This refers to the inaccuracy a finger can have on a screen when touching to interact with it. Compared to a mouse which has the possibility of pixel perfect precision when clicking anywhere on the screen a finger touches a general area in which the software has to determine the touchpoint which may or may not be where the user intended to press the screen. One solution to this problem is to make clickable screen elements sufficiently sized which depending on the exact implementation ranges between a physical size of 0.9cm - 1.6cm [21]. With larger individual screen elements less elements in total can be displayed at once and therefore constrains the information presented to the user as compared to keeping small screen elements.

This introduces the need for prioritization of what functionality that should be presented to the user. The most frequently used functionality should be easily accessible to the user to improve the usability of the software. But just as important are the functions that are infrequently used but essential for the program. Researchers in a recent study coined the term “raessential” for this type of functions [22]. It was shown that users on mobile platforms tend to be strongly attracted to corner icons in which the raessential functions can be guided to [22]. Another tendency for the mobile user is to search for functions in a depth-first manner, which was assumed to be due to memory load. The authors made the recommendation to provide visual cues to promote the user to utilize a breadth-first search method when the user fails to find the function they were looking for [22].

2.2.5 Interactions in challenging environments

Research has shown that challenging environments can affect the interactions with portable devices in a negative way. In the case of outdoor usage, the specific design of the screen itself can be a limiting factor for usability. For usage in high glare environments a glossy type screen ends up reflecting the incoming light from the sun making the screen difficult to read and navigate on [17]. Another contextual impact is that of the usage of a touchscreen during rainfall which both inhibits the vision of the screen and decreases the ability of the user to interact and navigate [23] with the screen. Furthermore, a study conducted on touchscreen usability in flight simulation showed a significant increase in task error rates and task completion times during turbulence compared to stable conditions [24], implying that users fail to mitigate the effects of unpredicted movement during interaction. Another key point in the study is that the addition of multimodal feedback based on auditory and visual cues did not reduce task error rate and slightly slowed the interaction down, but these cues were preferred by users compared to no auditory or visual feedback [24]. This implies that the user’s best perceived solution of the software can differ from the most effective one in practice.
In a study of GUI design of a military communications system for challenging environments [25], the researchers had to face the task of creating an interface for an application which was used in conditions such as moving vehicles, cold weather and with the support of devices with limited screen size and touch capability. They found that knowing the application usage in depth is central for good usability and creating a GUI which suited all situations. They had the following three different focus areas of usability: general usability, usability in challenging environments and usability with different devices. This led to the implementation of two separate user modes and GUIs depending on how the software was to be used, one basic mode for normal operation which was designed for a hand-held device and an expert mode designed for administrative use and advanced features. The expert mode had reduced usability and availability in hand-held mode due to the constraints of user input on the platform.

The GUI in the study was created with a hierarchical structure in mind having the central functions at the top level of the GUI and the more intricate less used functions within menus obscured from the front view [25]. The authors evaluated the GUI by having users do tasks with the software and then measuring the task-completion rate. They found that the users overall successfully completed most of the tasks. The authors found that most of the mistakes made by the users in the GUI were related to users losing themselves in the hierarchical structure of the design. This led the users to fail a task by not finding the action within the GUI, which was then followed up by failing subsequent tasks that were based on the same place of the GUI [25]. The study shares great similarity to the task in this thesis, to design the GUI of a military communications system used in challenging environments. The quantitative approach of the study gave data that showed that the design to a certain degree worked for the users. Although the study did not capture the qualitative approach with the contextual challenges and how to mitigate them with a GUI design as this thesis aims to do.
Chapter 3

Method

For this project a method based on the UCD-process was used. As described in section “2.1 User-centered design process” the process can be separated into four phases, each representing a vital part in creating a design that is satisfactory for the users. Each phase contains methods to reach the goal of the phase and to incrementally step forward in the process, with some phases being iteratively repeated until the process is completed. This chapter will describe in detail how the methods within the UCD-process were carried out in order to redesign and evaluate a new GUI for the PC-Dart software adapted for usage in challenging environments. Figure 3.1 outlines the phases of the process as well the iterations and method used in each phase.

![Figure 3.1: The UCD-process and methods used](image)

The first phase consisted of contextual interviews in vehicles. For the second phase, the data from the contextual interviews was collected and an affinity diagram was constructed and analyzed. From the data analyzed the third phase could be initiated with prototyping and a first iteration of a developed design. This design was then evaluated in the fourth phase of the process with users out of context with past experience of using the PC-Dart software. From this feedback the third phase of the process was repeated, and a final design was developed. This design was then evaluated in depth by current PC-Dart users in a military context. From this evaluation guidelines for future development could be drawn. Furthermore, general guidelines for designing for challenging environments were established using the data from all parts of the process.

3.1 Contextual interviews in vehicles

To start the first phase of the UCD-process and to reach an understanding of how the PC-dart software is used in its context of challenging environments contextual
interviews were conducted. The interviews involved five users (further on called U1, U2, U3, U4, U5) with experience of using PC-Dart in its context of challenging environments. All five users were currently using or in direct connection to the workflow of the PC-Dart software. The interviews were conducted during the army exercise Northern Wind in Norrbotten, Sweden [26]. This opportunity for interview and location was chosen to get the most accurate and fair representation of the usage of the program as possible. The interviews took place in the two different vehicles Eldledningspansarbandvagn (Epbv) 90 and Stridsledningspansarbandvagn (Stripbv) 90, which both utilize the PC-dart software. Two of the users U2 and U3 were at the time of the interviews directly operating the PC-dart software while U1, U4, U5 were working on supplementary work in and beside the vehicles in collaboration with U2 and U3.

The main point of the interview was to capture as much detail and information about how the user perceived, used and navigated in the software as possible in its proper context. The primary method for questions asked during the interviews was to observe the operator interacting with the software and inquire about the actions taken and reasoning for them. Furthermore, questions about contextual variations which were not present during the time of the interview such as work in challenging weather conditions or operation in nighttime, were initiated by the interviewer. During the interview statements and observations from the users were noted in bullet points for easy separation and categorization of data. The in-context interviews lasted for an approximate 90 minutes, in which the bulk of the data was gathered with the exception of minor general statements from U1 retold before arriving to the position of the vehicles. The interviews were not recorded since the vehicles were operating and running at the time of the interviews which made a recording without overpowering background noise impossible. For safety reasons and due to the circumstances of the exercise at the moment of the interviews, the vehicles in which the users worked were at a standstill.
3.2 Affinity diagram

For the second phase of the process an affinity diagram as described in section “2.1.2 Data interpretation and the affinity diagram” was constructed. The data from the interviews were separated into short notes, each one containing one statement/bullet point from the interview. The process of starting the diagram was to pick one note and separate it from the rest. Another note was then separated from the rest. If the next note was of a related topic to the previous note or any note already picked it was placed right next to it in a grouping. This process was then repeated with every note with the goal of keeping each grouping general or specific enough so that between 2-5 separate statements was in each grouping. Above these groupings, subcategories describing the statements were determined. Further above these subcategories top level categories were formed of groupings of the subcategories, again specific to the point of 2-5 subcategories within a top-level category (see figure 2.2, page 8). All the categories were formed and determined based on the statements and categories below them in the diagram. The complete diagram consisted 51 notes/statements which was combined into 15 subcategories and finally combined into five top categories. The diagram was then used to identify problems that the users conveyed during the interviews as well as specifying requirements and possible solutions to existing pain points in the current design that could be implemented in a new GUI.

3.3 Prototyping and implementation of the design

This section describes the process, software and hardware used in order to create a new design for PC-Dart for challenging environments. The first section “3.3.1 Prototyping” will describe the method used for creating a prototype of the design, the second section “3.3.1 Development and implementation of the design” describes the development of the design implemented for the destined target platform as well as the hardware and software used for the designs and evaluations.

3.3.1 Prototyping

The third phase of the process was initiated by creating rough concept paper sketches of a design. These low-fi sketches were then reiterated and created in the wireframing software Balsamiq [27]. In this software the design was created with partial interactivity with several different available views and including key elements such as message creation, message-inboxes and settings. The wireframe design was also scaled to target the screen size for the future implementation of the design to ensure that the differences in scaling between prototype to implementation was as small as possible. The design choices for the GUI were taken based upon the analysis of the data from the interviews further described in section “4.2 Identified problems/Proposed solutions with the GUI” as well as from the research on the topic in the section “2.2 Contextual HCI”. During the
process of prototyping and creation of the wireframe the design was discussed and evaluated continuously by stakeholders and previous users of the software out of military context.

### 3.3.2 Development and implementation of the design

The prototype served as a template for the design of the GUI of the developed implementation. The original PC-Dart software is a Windows programme coded over a time span of 20 years, written in C and C++ code. A new program shell had been created including some of the functionality of the original PC-Dart including connections to the original code libraries, radio and IP interfaces. As this thesis is aimed towards the GUI of the software and not the backend code, a continuation of this program shell was done. This limited the choices of development methods and language. The latest windows platform development method of UWP (Universal Windows Platform) apps [28] which caters to multiplatform deployment and touch enabled interfaces could not be used due to incompatibility with the original code libraries from PC-Dart. Therefore, the development was done in the older windows development platform of WPF (Windows Presentation Format) [29], which is aimed towards the windows desktop platform. The code for the design was written in C# and the design layout of the GUI in XAML markup. The developed design was targeted to a screen size of 10 inches and 1024x768 resolution to mimic the platform that the operators in the interviews described in “3.1 Contextual interviews in vehicles” used. The computer used for deployment and evaluation of the design was Microsoft Surface Go [30] as it had a similar form factor as the hardware currently used. A rugged tablet case was attached to the computer to further increase likeliness to the hardware used.

### 3.4 Evaluation of preliminary developed design

For the first transition from the third phase of development to the fourth phase of evaluation, the developed design and the process of its creation was presented in an open group out of field context. There were mixed background among the evaluators with most having military experience, some with direct experience of using PC-Dart extensively. The evaluation was done by presenting the GUI and the work done along with the key changes of the design compared to the original GUI to the group of evaluators. The evaluators discussed the design in an open group setting and described what they thought of the design as well as points of improvement, the evaluation session lasted for approximately ~45min. As this session was out of context and at a distance from the current usage it served mostly as gathering feedback on the general design to reveal potential missing elements in the GUI as well as thoughts on the design. The feedback from this session was taken into consideration and a new iteration of the third phase of the process of development was initiated.
3.5 Evaluation of the final design

In preparation for the final evaluation of the design and the last iteration of the fourth phase of the process, the feedback from the users in the evaluation described in section “3.4 Evaluation of preliminary design” was used to reiterate a new design. The final evaluation was conducted in the form of a usability test, think-aloud evaluation and interview to allow the users to get a deep understanding of the design at hand. The users got introduced to the purpose of the interview and was placed in front of the tablet running the new design of PC-Dart. The users were free to navigate in the software and were told to describe their impressions and thoughts of the design. Furthermore, questions were asked to the users in order to elaborate further on their thoughts of the design. The interview involved one current user and operator of PC-Dart referred to as O1 and one user referred to as O2 with experience of using PC-Dart in context as well as teaching future operators on the software. The interviews were conducted in the military compound of the ”Life Guards”-regiment [31]. As the purpose of the interview was to dissect the new GUI design in a qualitative manner and capture the fine details the interviews were yet again not done in moving vehicles but rather in a conference room at the regiment. The full interview was conducted in approximately 70min and was recorded in audio and later transcribed for analysis of the evaluation.
Chapter 4

Results

This chapter describes and goes into detail of the results yielded from the different phases of the UCD-process and the outcome of the process done to design a new GUI for PC-Dart adapted for usage in challenging environments. The section “4.1 Analysis of contextual interview and Affinity diagram” describes the results and data gathered from the contextual interviews and the first phase of the process. The section “4.2 Identified problems/Proposed solutions with the GUI” describes the results of the second phase of process, setting the requirements for the design. The section “4.3 Redesign of GUI” describes the results from different iterations of the third phase of prototyping, development and the different iterations of phase four of evaluation of the designs.

4.1 Analysis of contextual interview and affinity diagram

The top-level notes of the affinity diagram were the following:

- The user’s perception of the software.
- The user’s context.
- GUI and interaction with the software.
- Hardware and input devices.
- The software’s functionality.

Each one of these categories captures different themes of the software and its usage. Below these top-level categories are more specific themes within the category and below them, at the bottom, the raw data statements gathered from the interviews. The diagram does not only feature quotes from the user as statements but also observations of the user’s context and details of it. The following chapter will go into detail of each top-level category of the diagram and the statements below it. The five interviewees/users will be named U1, U2, U3, U4, U5 without any specific order in the following chapter. For the full affinity diagram see Appendix A.

4.1.1 The user’s perception of the software

The diagram category, “The user’s perception of the software” consisted of notes that described what the users thought and how they perceived the software. This category only contained a singular subcategory “User’s thoughts about PC-dart” as it differs from the other categories enough to deserve its own grouping. The consensus derived from the statements of U1, U2, U3 were that the software worked for its purpose despite that it had its flaws. U1 stated that “The workflow
with PC-dart is very uneven, sometimes there is very much to do at the same time and at other times there is nothing to do”, this description is very similar to the identified characteristics in chapter “2.2 Contextual HCI” expressing the need for high-speed interactions within the software.

### 4.1.2 The user’s context

The diagram category, “The user’s context” contained notes and statements regarding the user’s context when using PC-dart. It consists of the following subcategories:

- The user’s workspace
- Using PC-dart in low light environments
- Using PC-dart in a moving vehicle
- Learning to use PC-dart

The first subcategory “The user’s workspace” contains notes describing the hardware situation for the users. The tablet computer was differently mounted within the vehicle depending on the other external hardware that was supposed to be used together with the tablet. This led to different setups of input devices used with PC-dart. In the vehicle corresponding to U2 the keyboard was mounted to the tablet and placed above the screen, whereas in the vehicle corresponding to U3, U4 the tablet was mounted upside down which prevented the keyboard from being attached to the tablet and had to be laid flat on the table area in front of the user below the tablet.

The subcategory “Using PC-dart in low light environments” includes statements from U2 and U5 in which U2 described the action of lowering the light intensity of the screen to reduce strain on the eyes when the rest of the surrounding environment was dark. U5 perceived the light interface of PC-dart as a detriment to the work as the environment with contrasting light made it difficult to see other things in the dark as the eyes could not adapt to the darkness.

Under the category “Using PC-dart in a moving vehicle” the users unanimously agreed that it was very difficult to interact with the PC-dart software during movement as the effects of the moving vehicle makes interaction with the input devices increasingly demanding. U2 with the tablet mounted keyboard stated that “I have to unmount the keyboard when the vehicle is moving to prevent it from shaking uncontrollably” and further that “I almost strictly use the keyboard if I am forced to use the program during movement as it is very hard to use the pen”. U2 further told that the screen elements tended to blur together when the vehicle and screen was shaking.

The last category “Learning to use PC-dart” contains the user’s statements on learning how to use the PC-dart software. U1 recollected about how easy it was to connect two computers and use the software in between them to practice using the software or looking up a feature you were unsure how to use. U1 further said that generally the way to learn using PC-Dart effectively is “learning by doing” it.
4.1.3 GUI and interaction with the software

The category “GUI and interaction with the software” features the following subcategories:

- External information that is input into the system
- Problems for user to complete its tasks
- The window structure of the GUI in PC-dart

The first subcategory “External information that is input into the system” contains statements of things related to the PC-dart software but yet was found external next to the system or beside it as a supplement. In U2’s vehicle a small strip of tape was placed at the side of the screen on the border. This tape had common shortcuts written down on it as well as DART-message numbers. As part of the work the PC-dart operators (U2, U3) got instructions from another person in the vehicle calling out information to send in DART-messages. U3 utilized a separate paper beside the tablet with address numbers for input to the system when needed.

The subcategory “Problems that hinder the user to complete its tasks” relates to notes of apparent interaction problems with the interface of the PC-dart software. The current operators U2 and U3 expressed that some of the screen elements of the GUI required precise aim and input to get the desired action, which led to miss clicks or the need for repeated actions to get the desired outcome. U2 and U3 expressed that they never used their hand/fingers to interact with the screen as it was imprecise to effectively navigate with. Through observation of U2 and U3 it was noticed that not all message information was visible to the operator and that the user had to make use of the horizontal scrollbars by scrolling to be able to see the entirety of the message. This was a precise interaction as the scrollbar is thin and bordering another window.

The last subcategory “The window structure of the GUI in PC-dart” contains statements about how the users adapted the interface within the software to fit their work. The users U2 and U3 had different approaches to this and used different layouts based on personal preferences. U2 prepared messages in the software and kept them open in the background until the messages should be sent all while U3 kept a separate container in which saved prepared messages was contained until clicked and sent.
4.1.4 Hardware and input devices

The category “Hardware and input devices” features the following subcategories:

- The hardware used
- The stylus pen

In the first subcategory “About the hardware used” the users expressed some doubts about the hardware in the vehicle. U1 expressed the opinion that regular laptops probably could be used and replaced as they broke down instead of developing expensive custom-made rugged laptops. The concern was that the long development time of the hardware leads to it being outdated the day it is ready for deployment. U3 and U4 stated the problem that the keyboard often broke in its connectors leaving it unusable, needing the user to resort to the on-screen keyboard. The second subcategory regarded the stylus used as an input device for the software. During the interview U2 demonstrated some difficulty in using the stylus, to complete some actions several clicks were needed as the accuracy of the stylus was lacking. U2 was also forced to calibrate the stylus during the interview to avoid unnecessary mistakes in using the program and retold that this was often needed in order to keep the accuracy of the stylus on an acceptable level.

4.1.5 The software’s functionality

The category “The software’s functionality” features the following subcategories:

- Address book in PC-dart
- Alarms in PC-dart
- The user’s programme preferences
- The settings of PC-dart
- Functionality that the user’s miss

The first subcategory “Address book in PC-dart” regards the specific function of setting predetermined receivers of a message by binding shortcuts to it. This function was used by all users as it reduced the number of inputs the operator had to write for each message. The users believed the address book itself should be expanded beyond the total nine slots as the need for more address combinations were needed. Another functionality of the software is alarms that could be set in the software. U1 said that the alarms were useful in periods when there was low work intensity as the alarm drew attention to the software again, but the alarms tended to be annoying in every other case as it would beep constantly when more messages are incoming. For the user’s program preferences some differences in how they preferred to interact with the software could be observed. U3 used shortcuts for formats more extensively mapping a button to a specific DART-format that was often used while U2 preferred to manually input and create each DART-message.
During the interviews it was noted that the users had no trouble using the settings or setting up the software. U2 said that the work required to change the settings within the software often, but that the settings always was within reach if needed. U1 noted that there was some functionality that worked in a way that left out information to the user, for example if a message with several anticipated acknowledgements was sent there was no way to determine who acknowledged the message, but just how many of the recipients acknowledged the message.

4.2 Identified problems/Proposed solutions with the GUI

With help from the data gathered from the interviews and the categories from the affinity diagram key issues with the current design were identified. These key issues and elements served as the basis for the redesign and implementation of the new GUI. The chapter will point to these issues and potential mitigations to the problems and its implication for the design.

4.2.1 The light theme of PC-dart

As described in section “4.1.2 The user’s context” the users had issues with the light interface of PC-dart as it strained the eyes during prolonged exposure in the low light context of “Eldledningspansarbandvagn 90”. U2 described that it could be somewhat mitigated by manually changing the light intensity of the screen although this reduced the overall visibility and readability of the screen. This workaround is also separated from the software itself. A possible solution to this design problem is to offer the users an optional “dark-mode” theme for the software, tailored for usage in low light environments. As the software is used in a varying context both during daytime in broad daylight and nighttime in darkness the themes could be changed on the fly as it suites the user.

4.2.2 Interaction problems with PC-dart

As described in section “4.2.1 The user’s context” interaction with the software during movement of the vehicle was perceived as very challenging compared to interaction with the software as a standstill. The negative aspects of the user’s context in moving vehicles such as shaking and a general unstable platform for the user is inevitable as it is the nature of the work. Further described in section “4.1.3 GUI and interaction with the software” and “4.1.4 Hardware and input devices” the users had some trouble and interruptions when interacting with the software, both in terms of the design of the GUI leading to unwanted inputs on the screen and the hardware with the stylus pen requiring constant calibration for precise input.

The external factors to the software with the context the equipment is used in as well as the stylus pen used is difficult to affect. Although some of the issues caused by the negative aspects can be mitigated or reduced by the GUI design. The elements of the GUI can be visualized with a more differentiable style to ease the
4.2.3 Touch-input adaptation of the software

As the PC-Dart software is designed for desktop usage with a keyboard and a mouse/stylus the possibility for touch interactions with the software is limited. As the users described in section “4.1.3 GUI and interaction with the software” only the stylus pen was used to interact with the screen since using the fingers was inexact. This was partly due to the hardware used for the software but also the fact that no specific action was taken to enable or ease the use of touch input for the software. Actions you expect to see with a mouse/keyboard input combination is not the same as for touch input or even stylus/keyboard input. Interactions such as:

- double clicking
- right clicking
- hovering over elements
- scrolling and
- incremental movement of the mouse

are all examples of actions that were not used or not used in the same way as on an ordinary touch device. PC-Dart relies on these types of interactions for functions in the software. These interactions with the software need to be either changed or reworked for the new GUI design to completely enable touch input. Touch enabling the GUI will also help to mitigate the navigation problems described in section “4.1.3 Interaction problems with PC-Dart” as less inputs was needed for completion of action. E.g. a double click on data grid to review a DART-message, can be redesigned into a single tap on the same point at the data grid which reduces the amount of inputs that can be incorrectly interpreted by the software.

4.2.4 Information presentation in the software

As the operator of the PC-Dart software is dependent on the information received through incoming DART-messages to perform its tasks, the information presentation serves a central role for a functioning workflow. As described in the section “4.1.3 GUI and interaction with the software” the information presented to the operator of the software was limited in some situations. This limitation led to a workflow in which more inputs and interactions with the software was needed in order to complete the tasks of the programme, and as previously described some errors and faults when interacting with the software is expected. As the amount of information displayed at once on the screen is heavily dependent on
the screen size, some prioritization of the content might be needed for an adaptation to a portable PC-Dart GUI. As screen elements such as menus, buttons and text will be scaled towards touch compatible sizes often bigger than the desktop counterparts there might be less space for remaining information presentation. For the new GUI design the amount of information available to the user should be maximized all while the needed interactions and inputs with the software should be minimized in order for the user to complete its tasks with more precision.

4.2.5 Personalization of the software

As described in sections “4.1.2 The user’s context”, “4.1.3 GUI and interaction with the software” and “4.1.5 The software’s functionality” the users of PC-Dart had some differences in their external factors such as the hardware setup and workspace and internal factors such as setting up the PC-Dart software in their own way and utilizing short commands individually. These factors enable the users to adapt the software to their personal preference which can be a double-edged sword- the users own preferences might lead to more efficient work but lessens the uniformity of the software. The users should be able to interact effectively with the software as delivered without spending time on creating their own intricate setup. The GUI should still contain some possibilities for customizations in the form of shortcuts as the use case of which DART-messages to send can differ between different types of operators.

4.3 Redesign of the GUI

In the following section the different prototypes and developed designs are described in detail. The design choices for the GUI will be described and illustrated in terms of key elements and notable features that differ from the preexisting design of PC-Dart. The section is separated into three different sections. The first section, “4.3.1 Design iteration 1 - Prototyping and wireframe” presents the prototypes and wireframes of the design, the section “4.3.2 Design iteration 2 - Feedback from users out of context” describes the results of porting the wireframe into developed design and the feedback received on the design out of context. The section “4.3.3 Design iteration 3 - Feedback from users in context” describes the resulting design iterated from the feedback of the out of context users.

4.3.1 Design iteration 1 - Prototyping and wireframe

First initial low-fi paper sketches were created (see Figure 4.1), mimicking the target platform size to visualize and get a feel for how a potential layout of the design could be carried out. This sketch served as ideation for the wireframe created.
The wireframe prototype of the new GUI design was created in the software Balsamiq [27] (see Figure 4.2). The most apparent change from the original GUI of PC-Dart envisioned in this design is the dark mode of the software. As the analysis of the data from the interviews revealed the flaw in the software in section “4.2.1 The light theme of PC-dart” this was integrated into the design. In order to mitigate some of the interaction problems described in “4.2.2 Interaction problems with PC-Dart” the buttons in the left most side and the top of the design are sized according to recommendations [21] described in the section “2.2.4 Designing for portable interaction”.

To enable for as much information as possible to be displayed and avoid horizontal scrolling of the message list containers, a horizontal layout was opted for compared to the standard side by side message containers in the original PC-Dart layout (see Figure 1.1, page 2). This was also done by creating a button in the upper right corner (see Figure 4.2) of the design that allowed for the top menu to be hidden to be able to display more messages at the same time while hiding the less frequently used functions in the menu.
The left hand side menu of the design is the shortcut container which holds frequently used DART-formats used by the operator, and by clicking the “FMT” button the operator gets the choice to pick a format from the full list available. This was done in order to reduce the amount of inputs needed for the operator to get to the action of sending messages. As the design of the GUI was supposed to share the functionality from the old PC-Dart many similarities between the two can be found although the new version is adapted for easier comprehension for the user. As described in section “3.3.1 Prototyping” the wireframe was discussed and evaluated continuously during the process in an out of military context.
4.3.2 Design iteration 2 - Feedback from users out of context

The following section covers both the description of the design and the changes made from the previous iteration as well as the feedback from the user evaluation described in section “3.4 Evaluation of preliminary developed design”.

4.3.2.1 Design iteration 2 - Description

The first design developed on the target platform, described in section “3.3.2 Development and implementation of the design” (see Figure 4.3) shares great similarity to the Prototype (see Figure 4.2), as the feedback received on the prototype were sparse in terms of possible changes. For additional figures of the design see Appendix B. Rescaling of buttons and removal of scrollbars are the most apparent visual changes of the GUI. Another change from prototype to development is the two top left buttons which changes the left side menu between free format choice and bookmarks/saved shortcuts, to enable more seamless transitions between the two. Another change is the bottom message container which serves as the message send queue, which in the prototype were separated as message send queue on the left side and OPM (Operator messages) on the right side. The button for menu hiding in the top right corner were yet to be implemented in this version of the software.

![Figure 4.3: Design iteration 2, dark mode start screen](image)

4.3.2.2 Design evaluation, users out of context

The users evaluating the new GUI design out of context believed that they could see a clear distinction between the old PC-Dart GUI and the new one which raised
questions about where content from the old software was found or if it was present at all. The structure of the GUI with a side menu with the format choice and shortcuts were well received by the users. The dark mode of the interface was generally well received and one user retold that he had been asking for a darker setting for the software for ages when he was an active operator of the software. Another user noted on the dark mode that it might not completely keep the user’s night vision as it contained white text and other brighter elements that could reduce this effect, and that this would be a good point of research if the goal was to provide a true night vision mode of the software.

One of the evaluators pointed out that the window for operator messages was not visible in the interface and that it serves a central part and would be needed to be visible in some way in the interface. Another thing that the users missed was an indication of what radio units that were in use and that were connected to the system, as this is information that is needed for the operator to get a confirmation that everything is setup properly for sending messages. The users thought that the function of applying live dynamic filters to each message container (see Figure 4.3) might not be needed as the operators of the software do not need such flexibility in filtering and could reach the function by the top menu buttons as needed. The users stated that it was not apparent what each message container actually contained in terms of messages as there was no window border as in the old PC-Dart that indicated the content of the window. In terms of message input, one user had the idea of presenting the message in a horizontal direction instead of vertical rows as in the old PC-Dart GUI design and the new GUI.

4.3.3 Design iteration 3 - Feedback from users in context

This section will cover both the description of the design and the changes made from the feedback received from the users as well as the feedback from the user evaluation described in section “3.5 Evaluation of the final design”.

4.3.3.1 Design iteration 3 - Description

For the further development of the design described in section “4.3.2.1 Design iteration 2 - Description” many of the key features of the GUI remained the same as it was well received by the users evaluating it. Figure 4.4 shows the new iteration of the design with its dark mode enabled, for additional figures of the design see Appendix C. The dark mode of the application was further investigated and redeveloped from the previous iteration of the design, in accordance to Material Design guidelines by Google [32]. This added consistency through the design in terms of different brightness of the colors picked depending on how far back the different elements of the GUI would be perceived. The custom input for filters for the message containers were removed and headers describing the content of the container was added, as the users noted as points of improvement. Borders around the different message containers were also added in order to provide visual separation and feedback in the interface. Another feature change in the interface was in the bottom left corner of the design. Added to this is the visibility of system time as the same feature exist in the original PC-dart GUI. In the same corner, information about what radio unit that currently is connected
was added as per request from the users of the evaluation. The free space below this was left in case of several radio units were to be connected. A button for swapping between dark and light mode was added to the top menu as well as the button for hiding the top menu as featured in the first design iteration. Other minor general details of the GUI styling of the interface such as rounded corners, change of text font for text outside of message containers and styling of the message creation window was done.

Figure 4.4: Design iteration 3, dark mode start screen
4.3.3.2 Design evaluation, users in context

In this following section the primary operator and interviewee as described in section “3.5 Evaluation of the final design” will be called O1 and the second user O2. The user O1’s instant reaction to the design and GUI was very positive. O1 stated that “These large icons are really nice, it would make the usage in the vehicles very easy compared to what we have now, it is difficult to press the buttons that you want to when they are so small in size”. O1 then went on and recognized the left hand menu as shortcut buttons for formats and expressed that it would frequently be used as they as operators seldom used more than four to five different formats regularly and that the other menu for picking between the rest of the formats would suffice for quick access to infrequently used formats such as specific report formats. O1 stated that “This is how I would like it to be, easy and accessible, message handling can be difficult in stressful situations”, referring to picking formats in the left-hand menu. O1 further stated that although the self-programmable shortcuts available in the old design of PC-Dart, you still have to remember what format you bind to what button and that displaying them on the screen in a menu at the same time eliminates that issue.

As O1 navigated through the format picker menu he got stuck clicking the “Tillbaka” (back)-button in the menu (see Figure 4.5), expecting that it would take him back to the shortcut screen but it took him back to the top level of the format picking screen. O1 further told that he could see other users doing the same thing and perhaps thinking the software froze as the button did not take them anywhere and try to restart the software as a counteraction.

![Figure 4.5: “back button” highlighted](image)

O1 expressed that they are very much dependent on shortcut keys such as CTRL+F for new format and CTRL+S for sending, as the buttons in the current GUI are difficult to click or reach when the vehicle is moving. O1 said that even though it would be easier to navigate through the new GUI design it should probably be possible to still interact with the software through shortcut commands as in the old GUI, as different operators wish to work in different ways. O2 also stated with his experience of leading and instructing users in PC-Dart that the users need some flexibility in using the software or customizing it just as they
have when they use their private computers or phones, just to keep things fresh and not completely linear. O2 wanted to be able to have flexibility in sizing the message containers in the GUI to be able to fit more information on the screen, as O2’s layout could consist of up to eight different message containers, perhaps by reducing font size as O2 believed he would be fine with much smaller text.

O1 recognized and found the button for the dark/light mode. O1 expressed the instant reaction upon reviewing the dark mode that it would probably be utilized all the time and not necessarily only in nighttime and other dark environments. O1 described that they as operators currently turn down the screen brightness to a minimum just as the users in the contextual interviews described that they did. Furthermore, O1 expressed that the solution of having a dark mode on the software would be much better than just adjusting the screen brightness and that it could even be used supplementary with the brightness adjusting. O2 also expressed that a dark mode would be especially useful in the field in observational spots as they must be as invisible as possible. Currently in such situations the operator has to cover themselves and the screen of PC-Dart and lower to minimum screen brightness to not be revealed in the dark environment, which then created the problem of the operator having poor vision in the dark once they were supposed to switch observation spot. O2 noted that the button for swapping the dark/light mode was in the start screen beside the setting button and that this placement might be problematic if you were to miss click the button setting up to potentially blind you in a dark environment or reveal your own position in an observational spot.

About the colorful borders around the message containers (see Figure 4.4), O2 stated: “It is a good addition, after 15 days in the forest you start to get a bit mentally fatigued, but once you learn what color that corresponds to each message container you might not need to even read the header of the container”. Both O1 and O2 missed the function of resetting the software, in the case of emergency, on the front screen, even if it is a seldom used function the consensus was that it is so vital that it should be present in a start screen but at the same time prompt you a number of times so that there is no risk of resetting the software unintentionally.

O2 expressed the interest in keeping logs of the usage of PC-Dart during exercises to be able to track what went wrong in the software and what parts that were not used at all, as it could help determine where the operators would need more or less training and how the design of the software could be developed for future use. O1 and O2 together brought up an idea of having different roles in the software, so that when you start up PC-Dart you get to pick what role you are using the software as, Forward artillery controller as an example. The different roles could have saved shortcuts, formats and settings that are appropriate for the role, reducing the setup time for the individual user.
Chapter 5

Discussion

In this chapter, the results of the evaluation of the design will be discussed and related to the research previously done on the topic. Following this, the methods used for the UCD-process will be discussed in terms of how they worked towards the goal of creating a design, and the limitations of the methods in this project. The design guidelines for the process of designing for challenging environments will be described as well as possible extensions of research of the area in the future work section.

5.1 Results

As the results of the contextual interviews and the analysis of it, the users could use the software to allow them to complete their tasks there were several points that could be improved upon to allow for easier and faster usage. To completely eliminate some of the issues described by the users a new hardware platform would probably be needed as its limitations of input accuracy could not be entirely eliminated by a new GUI design, which is beyond the delimitations of this thesis.

The aspects that could be affected by GUI design leading to a better way of interaction and overall usage with the PC-Dart software had great potential according to the user’s response to the design described in section “4.3.3.2 Design evaluation, users in context”. As the contextual data pointed to the PC-Dart software was in need of a dark theme for the software to be able to use the program to its full potential in its context of usage in dark environments, and the post development interview confirmed that it was a solid design choice that would be used extensively if deployed on the software and devices in use. Another substantial aspect that the observational data from the contextual interviews revealed was the poor accuracy in the software paired with small elements in the GUI leading to miss clicks within the software. This led to the design choice of making all elements a certain minimum size as per recommendation of previous research [21]. The users’ evaluation of the new GUI design was positive in the manner that the design would help them interact with the software and easier do their daily work with the contextual challenges present.

The users in the post development evaluation described possible extensions of the design. This includes role based adaptation of the software much like the one described in the previous research in section “2.2.3 Responsive and adaptive user interfaces” [19,20], and possible logging capabilities that the previous research used for evaluation of their design [25] described in section “2.2.5 Interactions in challenging environments”. These features were not revealed as needed/wanted by the users in the initial contextual interviews and were therefore not prioritized for the developed design. This confers to the advantages of using an iterative
process in creating a new design. The users get a chance to reveal flaws that are not yet apparent to the designer, and at the same time confirm that data from other research applies in the same manner to the specific project in question.

For the iteration of the design from wireframe to the first functional developed design as described in section “4.3.2.1 Design iteration 2 - Description”, no formal evaluation was conducted, only continuous feedback received during the process was taken into mind. This led to small changes between the designs. The platform for development also contributed to the changes for the initial design. Details such as a functional search input field for the format picker menu (see figure 4.2) was deemed to take time from other more important details of development, as well as having limited use or improvement for the function of the design. Another change between the iterations which turned out to improve the design was having the format message name on the buttons in the format picker menu. This provided additional visual feedback to the user leaving less to memorize in terms of formats as the user O2 described in the final evaluation. This can be related to the research in section “2.2.5 Interactions in challenging environments” in which the users felt aided from added visual and auditory cues in the software when navigating [24].

For the second iteration of the developed design described in “4.3.3.1 Design iteration 3 - Description”, the feedback received by users out of context was used to improve on areas of the design that were extra notable to said users. The main point of feedback received on the design involved functions that the users felt was necessary to the software yet was completely missing in the design. Such as the operator messages and status of the connected radio units, as described in “4.3.2.2 Design evaluation, users out of context”. As a designer inexperienced with the software at hand, such details can be easily overseen or forgotten about in a preliminary design. As functions such as the above mentioned can seem insignificant when testing the functionality out of context yet is completely necessary for the user in context usage and environment. This further speaks for the need of user involvement in the design and an iterative process.

The context of the user effects the user’s interaction with the software all while the software’s GUI affects the user interaction as well. Therefore, a need to approach the development from both perspectives is needed, adapting the GUI towards the specific context of the situation. This thesis incorporates this methodology leading to the results described above. In general, applying this methodology when approaching the user experience in software used in challenging environments could lead to a more satisfactory result than approaching the user experience from a single perspective. When the focus of the process lies on the user from all point of views, a properly executed process will lead to improvements for the user in their context.
5.2 Methodology

The methodology of the thesis with a UCD-process to design for the contextual intricacies of challenging environments generally worked well, with some exceptions and weak points. As described in the related work done in section “2.2.1 Military HCI”, close work with users in a military context is often difficult to arrange in earlier stages of a project [17]. A lack of representative users involved in the process at an early stage or even all stages of the process could make the results diverge from the user’s true needs. In this process users were involved close with the project but only on limited occasions and in small numbers, as mentioned in the section “1.4 Delimitations”.

The method of contextual interviews worked well in the process and gave a great deal of contextual data that otherwise would be difficult to identify as a non-user of the software. If the interviews were conducted out of context there would be a risk of not observing the user under the real circumstances they are facing. In this case, the user’s workspace would not be present and how the user adapts to the limitations this brings. Furthermore, other cognitive disturbances such as loud engine noise and vibrations effecting the user would be removed. The method captures the context and the user in an extensive way. Of the data captured there was much related in some way to the specific goal of creating a new GUI design, although some data pointed primarily towards the hardware platform. This makes for difficult distinctions of what aspects that were limited by the GUI of the software or simply the hardware. If the goal would have been to take a full system solution from software to hardware in mind the method of contextual inquiry would have sufficed to capture all aspects.

The affinity diagram and the data analysis of it served as a tool to take all the data from the contextual interviews and breaking it down into aspects that concrete design choices and requirements can be drawn upon. It is mentioned in the description of the method that it preferably should be done in a cross-functional team that you would usually have when creating a design [8]. Bringing more diverse perspectives into the method might help the categorization as it is less dependent on an individual for decisions. The same goes for the analysis of the diagram and the outcome of the analysis. With this said, with careful analysis and categorizing the method works for bringing out themes in the data in a good manner.

For evaluation of the prototype and the developed versions of the design it was as previously mentioned in the section difficult to get more users to test the design. This fact prevented to test the design in a quantitative manner, as done in the research described in section “2.2.5 Interactions in challenging environments” [25]. For the purpose of this study to design for challenging environments, the qualitative approach with a few selected users for interviews gave enough data to believe that the thoughts and opinions expressed of the users could apply to the user base operating the PC-Dart software under similar conditions of the users interviewed. As the thesis focused on a subset of the use case and user groups of
the software it is difficult to determine if users with completely different tasks would yield the same results when evaluating the design. It is important to notice that having more users to interview and evaluate the design on probably would reveal more valuable data and opinions of the design. Furthermore, to catch the finer details of the contextual challenges at hand, logging the usage of the software as described as a possible functionality by O2 in section “4.3.3.2 Design evaluation, users in context”, would probably be a great addition.

5.3 Design guidelines

Designing GUIs for usage in challenging environments and contexts adds an additional layer of complexity to consider in the design process. The contextual challenges the users face in their usage of a system can vary depending on what the system is used for and when the system is used. The following points describes the design guidelines when designing GUIs for usage in challenging environments:

- **Establish the context in which the design will be used, and what challenges the context will entail on the user.** A question to ask is if the user uses any other system with similar contextual circumstances and if the interaction is limited in these cases. An example of this could be the case where a user needs to input data during continuous movement, which at first glance seems like a hard task, but shares great similarity with the interaction of texting with a mobile phone which is a task that many do in their everyday lives.

- **Work as close as possible with the users, involving them in the process.** Direct user involvement in the design process as much as possible is also advised in order to really grasp how the users work with the system as well as getting valuable feedback to further develop the design towards satisfactory results. An important detail to keep in mind is that users are good at learning and adapting their usage towards what the design allows. This means that as a designer you have to not only look at what the user is doing, but rather what the user should not have to do in order to complete their work. Just because something is currently working with a system for the user does not mean that the experience for the user cannot be improved upon, especially in the context of challenging environments.

- **One solution might not fit all contextual situations or users.** In some cases, in where the software has a broad usage with different contextual situations and user with different needs, a single adaptation of the GUI might not be enough to be satisfactory to everyone. Utilizing different optional settings or GUI adaptations such as described in the research [19, 20, 25] can be critical for the user experience. Something as simple as preloading settings in roles for different user groups can improve the experience for the users with the software.
• **Apply a full systems approach to capture all contextual challenges.** Just as important that the GUI design is to enable interaction for the user with the software, a broader system approach is needed to create the best experience for the user. Co-designing the software and the GUI together with the hardware to ensure compatibility between them is just as important for the user experience. Another additional step can be taken in the process to ensure that proper education to the users of the software and hardware is included and accounted for in the design.

### 5.4 Future work

For future work into the topic of contextually challenging environments the specific conditions of the context could be researched further. The topic of adapting a GUI to be used in a low light/dark environment could be researched further. Perspectives to take in mind would be to not only from a user perceived reduced strain on the eyes, but to research with the goal of defining a documented solution for keeping the user’s eyes adapted to the darkness. The design outcome of this project featured a dark mode, but without tests in a dark environment to measure the performance of it. This topic could be an interesting continuation of research. To further develop the adaptation of the GUI and software for different challenging contextual cues and user groups is a topic that could be expanded upon in further research. The objective of this research could be to aim towards creating a seamless transition of the GUI for the environment at hand. Another topic of challenging environments would be to take the hardware design into focus as well, a full systems development approach on how to design hardware paired with software for usage in challenging environments. This to achieve the best possible working environment despite the challenging context.
Chapter 6

Conclusion

The stated research question for the thesis was:
“In which ways can a redesign of an existing GUI based on user-centered design methods reduce the impact of challenging environments on usability when using an application on portable platforms?”

This is answered in detail by the results of the evaluation of the developed design in section “4.3.3.2 Design evaluation, users in context”. The users involved throughout the UCD-process in both the contextual interviews conducted and the design evaluation out of context described specific problems related to the challenging context. This data was evaluated in which a redesign of the GUI could be targeted towards. The suggested redesign developed out of the process would according to the users in an in context evaluation, make the work done by the users more enjoyable, less error prone and more qualitative.


### Appendix A - Affinity Diagram

<table>
<thead>
<tr>
<th>The user's perception of PC-dart</th>
<th>The users context</th>
<th>GUI and interaction with PC-dart</th>
<th>Hardware and input devices</th>
<th>PC-darts functionality</th>
<th>Functions that the users miss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users thoughts about PC-dart</td>
<td></td>
<td>Problems for user to complete its tasks</td>
<td>The design of the GUI in PC-dart</td>
<td>Address book in PC-dart</td>
<td>The users programmed functions</td>
</tr>
<tr>
<td>The users workspace</td>
<td></td>
<td>User interface complexity and aesthetics</td>
<td>The styles panel</td>
<td>Alarms in PC-dart</td>
<td>The settings of PC-dart</td>
</tr>
<tr>
<td>Using PC-dart in low light enviroments</td>
<td></td>
<td>User interface complexity and aesthetics</td>
<td>The styles panel</td>
<td>The users programmed functions</td>
<td>Functions that the users miss</td>
</tr>
<tr>
<td>Using PC-dart in a moving vehicle</td>
<td></td>
<td>User interface complexity and aesthetics</td>
<td>The styles panel</td>
<td>The users programmed functions</td>
<td>Functions that the users miss</td>
</tr>
<tr>
<td>Learning to use PC-dart</td>
<td></td>
<td>User interface complexity and aesthetics</td>
<td>The styles panel</td>
<td>The users programmed functions</td>
<td>Functions that the users miss</td>
</tr>
</tbody>
</table>

- **Appendix A**
- **Affinity Diagram**
Appendix B - Design iteration 2
Appendix C - Design iteration 3

<table>
<thead>
<tr>
<th>0 --</th>
<th>Radioformat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 --</td>
<td>Ledning</td>
</tr>
<tr>
<td>2 --</td>
<td>Luftförsvar</td>
</tr>
<tr>
<td>3 --</td>
<td>Sjömål</td>
</tr>
<tr>
<td>4 --</td>
<td>Understöd</td>
</tr>
<tr>
<td>5 --</td>
<td>Und-tjänst</td>
</tr>
<tr>
<td>6 --</td>
<td>UTMUnderstöd</td>
</tr>
</tbody>
</table>

### Huvudregister - Mottagna

<table>
<thead>
<tr>
<th>Tid</th>
<th>Till</th>
<th>Frlm</th>
<th>Ufrlmt</th>
<th>FMT</th>
<th>KVI</th>
<th>Annm</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>200923</td>
<td>100</td>
<td>0</td>
<td>FRI<em>TEXT</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>231507</td>
<td>600</td>
<td>0</td>
<td>ETI<em>BEGÄRAN</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>231507</td>
<td>104</td>
<td>0</td>
<td>KVITTENS*</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>231508</td>
<td>100</td>
<td>0</td>
<td>FRI<em>TEXT</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>231508</td>
<td>600</td>
<td>0</td>
<td>ETI<em>BEGÄRAN</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>231537</td>
<td>400</td>
<td>0</td>
<td>ETI<em>BEGÄRAN</em></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Utgående

| Tid | Till | Frlm | Ufrlmt | FMT | KVI | Annm | Format    |

---

![Image of DART settings window]