Delft University of Technology

Final Report

Search and rescue Awareness and Information Tool (SAInT)

Christian Vermorken [1358499]
Alexander Dirkzwager [1397834]
Sebastiaan Meijer [1374168]

29th June 2010

Final Report

BSc-project USAR:

Search and rescue Awareness and Information Tool (SAInT)

IN3405

Christian Vermorken [1358499]
Alexander Dirkzwager [1397834]
Sebastiaan Meijer [1374168]

Company:	TNO Defence, Security and Safety
University:	Delft University of Technology
Faculty:	EEMCS, Man-Machine Interaction Group
Employer:	Marc Grootjen
TU counsellor:	Tjerk de Greef
Coordinator BSc:	Peter van Nieuwenhuizen

29th June 2010

Preface

This BSc project is the final phase of the Computer Science bachelor education. The goal of the this project is for students to experience the entire process of software development from the design to the implementation phase.

During the past three months we have been working in a team of three people on the "Collaboration at a Distance" project, a joint effort between TNO Defence, Security and Safety and the Delft University of Technology.

We would like to thank our employer Ir. Marc Grootjen and Drs. ing. Tjerk de Greef our TU counsellor for their assistance during the project. Additionally, we would also like to thank Ir. Yusong Pang and Ir. Marc de Hoogh who kindly assisted us with RFID technology and gesture recognition.

Contents

Executive Summary

The Search and rescue Awareness Information Tool (SAInT) project is part of a larger project called "Collaboration at a Distance", a joint effort between the Delft University of Technology and TNO Defence, Security and Safety. The goal of this project is to identify and create tools to improve the communication over distances for the USAR.nl organisation, a special rescue team that will be dispatched to disaster areas.

The system has been designed using the Situated Cognitive Engineering (sCE) method, consisting of three different phases. In the first phase a domain and support analysis has been done, to identify operational demands, human factor issues and technological possibilities. The second phase consists of identification of the SAInT requirements of our system. In the third phase a prototype has been developed, consisting of three different tools. This is a iterative process where requirements are specified and prototypes are made and reviewed. Based on these reviews, the requirements are refined.

We have developed three different tools. One of these tools is an iPhone application, which will be used by the commander of a search and rescue team. This application will allow him or her to enter and view information about victims and dangerous areas on a map. Detailed information such as text and photos can also be added to these victims and dangers. Another tool has been created on the iPad, which is use by the operational commander and a technical worker of the support group. The commander is provided with the information entered on the iPhone, as well as statistics of each search and rescue team. The technical worker has access to detailed information about equipment. Each piece of equipment can be uniquely identified by its RFID-tag. The third developed tool, an RFID-application, uses these RFID-tags. This tool can be used to checked items in and out of the base camp. This can be seen on the iPad, to track the movement of equipment during a mission.

The created systems could still be improved by adding more functionality focused on communication between different members of the organisation, such as a message system, or adding new options for equipment management. This will decrease the dependence on older methods or less optimal systems.

Chapter 1

Introduction

This bachelor's project is part of the "Collaboration at a Distance" project, which is a joint effort between TNO Defense, Security and Safety and the Delft University of Technology. This project operates in the Dutch "Urban Search and Rescue" domain and aims to improve the collaboration between people working at different locations. USAR.nl is usually deployed after a disaster and have to be as independent as possible. It is possible for infrastructures to be destroyed and local authorities to be overloaded, therefore the team has to been an entire, self-sustaining organization bringing their own equipment. After arrival, a base camp is built, the staff tent is set up and several rescue teams deployed over the disaster area to search for survivors. The operational command group is usually travelling between the base camp and local, regional, or national organisations. Furthermore, teams in the Netherlands are taking care of support and coordination of, for example transportation of personnel, equipment, and other supplies to the disaster area.

Since most of the USAR.NL teams are so widely spread over the disaster area, it is very difficult to communicate, gather, and distribute information, which, obviously, is essential to the success of the rescue missions. This project will design Human Computer Interaction tools that improve the efficiency and effectiveness of the urban search and rescue teams.

Problem Definition

During this project, we will be developing a prototype system that will be able to gather, save, and present essential information to the rescue workers in a effective manner, ensuring that the right people get the right and relevant information. In order to achieve this, we will study the possibilities of using different technologies and platforms.

Goal The goal of current project is to improve the information exchange, by making the information not only easier and faster to interpret, but also simpler to collect, save and present. This makes for a much more effective and efficient way of distributing information. This information exchange allows for the rescue workers to be better informed, and thus giving them better situational awareness. This makes the USAR teams more effective and efficient.

This project has three main focuses (more details in Appendix B section 2.4):

- Information input
- Automatic gathering of information
- Information presentation

Situated Cognitive Engineering Methodology

During this project we followed a relatively new method, Situated Cognitive Engineering (sCE), developed by Neerincx et al. This method "follows a theory- and empirical-driven design process" and "is highly iterative with incremental top-down developments of functions." [7]. A full description of the method falls outside the scope of this report, for more information about the method see paper by Neerincx et al [7]. We will briefly explain the three main phases in the sCE method, *Derive, Specify, and Test and Refine*, which can be seen in Figure 1.1.

Derive In this phase, a work domain and support analysis is performed, focusing on the operational demands, human factors and envisioned technology.

Specify In this phase, the requirements are specified based on the analysis. Claims, (justifications) which are derived from the analysis are also. Use cases are made as well to contextualize the requirements.

Test and Refine In this phase, the system is evaluated through reviews and simulations. Prototypes are made, tested and reviewed. Finally the requirements and products are refined.

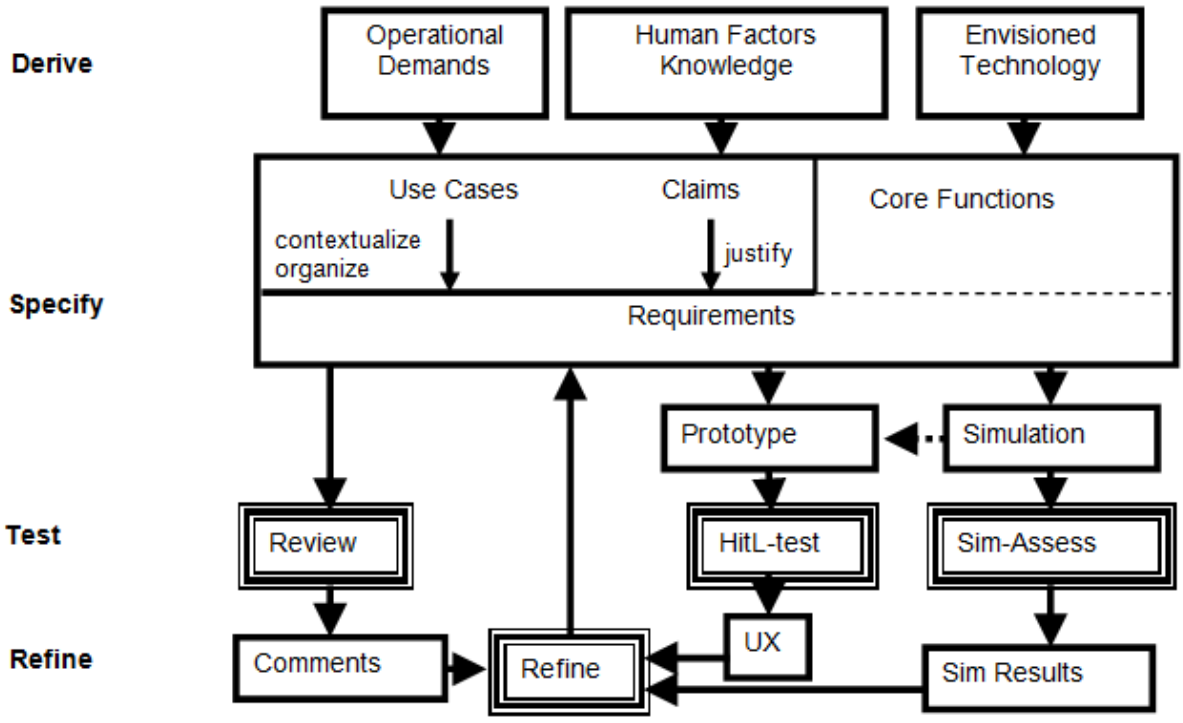

Figure 1.1: The Situated Cognitive Engineering method process.

Reading Guide

This document is organized as follows:

Chapter 2	Analysis of the USAR work Domain
Chapter 3	Design of our System, Requirements, Use Case and State Diagrams
Chapter 4	Implementation Process
Chapter 5	Conclusion
Chapter 6	Recommendations
Chapter 7	Reflection
Appendix A	Project Description
Appendix B	Report Orientation Phase
Appendix C	Requirement Analysis Document
Appendix D	Design Document
Appendix E	Test and Implementation Plan
Appendix F	Gesture Recognition

The appendices were previously written in the first couple of weeks and will be summarised in chapters 2 and 3. Chapter 4 will describe the implementation process and any changes made to the original design. Chapter 6 contains the recommendations for future work on this project. Finally, chapter 7 reflects on the entire project, as well as the sCE methodology.

Chapter 2

Analysis

In this chapter, we will present the results of an analysis of the problem, conducted by means of "Situated Cognitive Engineering", a method developed by the "Collaboration at a distance" research group at the TU Delft and TNO. A full analysis of the domain can be found in the Requirements Analysis Document in Appendix C.

We have conducted an analysis of the problem using the "Situated Cognitive Engineering" (sCE) method. This method was developed by Neerincx et al. [7] to acquire the requirements for a software project, with a strong focus on the human aspects of a software engineering project. Figure 1.1 shows the process of deriving and refining the requirements.

We started analysing the domain by interviewing Marc Grootjen and Tjerk de Greef, who have been observing a training mission of the USAR.NL-team in the Czech Republic, and have been working on the "Collaboration at a distance"-project for many years. Interviewing members of the USAR.NL-team was not possible within the short time span of this project.

We have interviewed these experts and read the research that has been done on the sCE-method [7], the field [4], and other projects in which the sCE-method has been used [6, 8]. We have also consulted reports, photos, videos and logs made during the training mission in the Czech Republic, and the last mission in Haiti. This way, we acquired sufficient knowledge to derive the operational demands, human factors knowledge, and envisioned technology.

Deriving the operational demands, human factors knowledge, and envisioned technology is an iterative process, in which we frequently discussed the results of our analysis with the experts in the field, and changed the analysis accordingly. The final result of this analysis is presented below.

2.1 Operational Demands

The analysis of the operational demands shows that the structure of the USAR organisation is hierarchical, consisting of different groups with clearly defined tasks. The USAR.NL team can deploy anywhere within 5000 kilometres from The Netherlands. When the USAR.NL team is deployed, they take everything they need for the search and rescue operation with them. On the disaster site, a base camp is set up, from which operations are coordinated. The four search and rescue teams operate autonomously, and on remote locations away from the base camp. This poses a challenge for coordinating effort to find and rescue victims quickly and efficiently.

Based on the analysis of the operational demands, we have formulated the following objectives:

1. The system will keep the commander and staff group informed by collecting and presenting information about personnel and activities, by enabling users to input information about

their activities, and by tracking movement of search and rescue teams

2. The system will keep the technical workers informed by collecting and presenting information about equipment, which is gathered with little user input

2.2 Human Factors Knowledge

The most prominent aspects of the Human Factors Knowledge that have to be taken into consideration for this project are worker competencies, collaboration, situational awareness, and time limitations.

Team members may need to work for long periods of time in extreme weather conditions. This could cause worker competencies to drop, which could lead to mistakes and errors. Collaboration between the different teams is also a critical factor, since the search and rescue teams, as well as the other teams that stay in the base camp, work independently, yet regularly need to request information from other teams. Since the situation at a work site is constantly changing, creating situational awareness is key to the success of the operation. A comprehensive overview of the situation is therefore needed, to create the situational awareness as quickly as possible, since time is a critical factor in rescuing trapped victims.

Based on the analysis of the human factors knowledge, we have formulated the following objectives:

1. The system will improve the collaboration at a distance between teams
2. The system will improve situational awareness when switching teams on a work site
3. The system must be easy to use
4. The system will provide feedback to the user when needed to increase the user's trust in the system

2.3 Envisioned Technology

This section contains an analysis of the possible technologies that can be used to solve the problem. Since different USAR team members have different information needs and work in different circumstances, it is not sufficient to create a single application on a single device.

The commanders of the search and rescue teams need a small and portable device that they can easily take with them to a work site, which can be interfaced with a spray can to capture annotations made on walls, and capture the location and the state of the victim. An iPhone could be used for this purpose, since it has the necessary hardware. A disadvantage of the iPhone would be the small amount of information that can be shown on the screen at once, and the fact that it cannot be operated while wearing normal gloves.

The operational commander needs a portable device, which has a large enough screen to provide good overview of the situation at the different work sites. An iPad could be used, since it is portable, has a 10.1 inch screen, and provides a similar interface as the iPhone. All alternative internet-tablets are still in development, so the iPad is the logical choice.

Research has been done (see Appendix F) on whether we could use the iPhone in conjunction with a spray can to recognize certain gestures. However, the gesture recognition technology is not yet advanced enough for that intended purpose, so here, technology is the limiting factor. In the future, when gesture recognition has been researched further, this idea could be expanded upon.

For the checking in and out of equipment, RFID-technology will be used. RFID-tags can be easily attached to tools and other equipment, which will make registration of equipment easy and fast. We will use passive RFID-tags, since these do not require a battery, are robust, have an acceptable cost and range, and do not require much maintenance. These tags are, however, susceptible to electromagnetic interference, and do not work well near metal objects. Also, there is no software available for other operating systems other than Microsoft Windows, so a Windows-computer with touch screen will be used instead of a more portable device such as an iPad.

Based on the analysis of the envisioned technology, we have formulated the following objectives:

1. An iPhone application will be created to aid the commanders of search and rescue teams
2. An iPad application will be created to aid the command group
3. RFID chips will be used to automatically collect information about personnel and equipment

Chapter 3

Design

This chapter will summarise the design made in the first weeks of the project. A more detailed overview can be found in the Requirements Analysis Document (Appendix C) and the Design Documents (Appendix D). The first section of this chapter will describe the system's core functions and the general idea behind the requirements of each of these functions. Then the non functional requirements will be brought to light. followed by the claims and use cases. Then something about the storyboards. In the section 6 we will show the architect of the systems through diagrams.

3.1 Core Functions, Requirements and Claims

During our requirements analysis, we have identified four core functions (Appendix C section 4.1) . Each core function is a high-level of the functionality our designed system would have. From these core functions we derived our functional requirements and for these requirements we developed claims to justify (Appendix C section 4.2). The claims as specified in the Situated Cognitive Engineering method are acquired using what was learnt from the analysis.

Automatically collect and present information about the location and status of equipment This core function mainly focusses on information gathering by using RFID-tags and scanners. Every piece of equipment will have a unique RFID-tag, used to identify it. Whenever equipment is checked in or out, several statistics are generated automatically. These statistics can then be viewed on the iPad.

Provide the users with the right information at the right moment The second core function mainly focusses presenting information to the user. Detailed information of victims and dangers can be viewed, as well as manually entered information about equipment. Requirements on manually inputting this information will be described in the fourth, final, core function.

Provide the users with a clear and easily accessible overview of the situation. The third core function mainly focusses on providing users with a map view, combined with symbols for each victim and each danger. This will provide all necessary information to a rescue worker, without cluttering the screen with more, less useful, information.

Allow manual input of information about victims or dangers at a work site, including detailed information Finally, we have identified a core function related to the manual

input of information. The requirements related to this function focus on manually inputting information about victims and dangers, as well as registering equipment and their status.

3.2 Non-Functional Requirements

In the Requirements Analysis Document (Appendix C) section 4.3 we also created non-functional requirements. Non-function requirements are requirements that have to do with the operation of the system instead of the behaviour or functions (which are the functional requirements). We specified our non-functional requirements in different criteria:

Reliability The system has to be reliable because if data is lost or inaccessible this will cause duplication of work. When time is of the essence as it is in a disaster situation this can be very costly, decreasing the chances of a trapped victim to be rescued by a USAR team.

Time Efficiency This system should be able to provide services that will make some of the processes more time efficient, for example when one team is relieving another team. With help from our system it will improve the exchange of information and increase the situational awareness of the commander.

User Interface and human Factors Our system needs to have a clear and simple user interface to allow for easy use, increasing the usability and learnability. Since our system consists of different platforms we want it to be easy to switch between platforms. Our system is also meant to be taken out into the field, so users should be able to use the system efficiently.

Hardware Considerations The system consists of different platforms so the hardware has to be taken into account.

Performance Characteristics For our system to perform effectively within the work domain, certain performance characteristics needed to be satisfied.

Error Handling and Extreme Conditions For the system to be reliable and robust it should be able to handle errors to prevent data loss.

System Interfacing The system should be able to communicate with the other platforms through a server.

Quality Issues Requirements that ensure that the code is of good quality.

System Modifications This system is to enhance/modify certain tasks or processes, it does not replace any process.

Physical Environment The iPhone application will be used on the work site, so certain requirements will need to be satisfied to be able to work in extreme conditions (in which the USAR teams work in).

Security issues Since informations is stored and retrieved from the server there has to be security measure to prevent unwanted access to the information on the server.

3.3 Use cases

In the Requirements Analysis Document (Appendix C) section 4.5 we provide numerous use cases. In the Situated Cognitive Engineering method use cases are used to contextualize the requirements. These use cases also provide information about certain situations where the system will be preforming in, they will show how our system will respond to an actors action within these situations.

The use cases are split up into situational categories:

- Checking the Equipment out
- View Map
- Adding Items to the Map
- View Detailed Information of an item
- Edit Map Item
- View statistics
- Identify a Piece of Equipment
- View Information about Equipment
- Update Equipment Catalogue
- Manage Maintenance Information

3.4 Storyboards

In the beginning of our implementation phase we made storyboards of our user interface to be able to present to our users to be able to get feedback (one of the iteration processes we went through while using the situated cognitive engineering method. The Storyboards can be found in the Appendix C section 5.

3.5 System Architecture

In the Design Document (Appendix D) we show varies diagrams that show the architecture of how we envisioned our system. During the implementation phase we did deviate to this some what. This will be discussed further in the next chapter.

Class diagrams

The following diagrams show how our system architecture looks like. We have created our applications in the paradigm of Model - View - Controller.

3.5.1 Model

1. **iPhone and iPad Model** The Model contains all classes that hold information. The Model bellow is shared between the iPhone, iPad application.

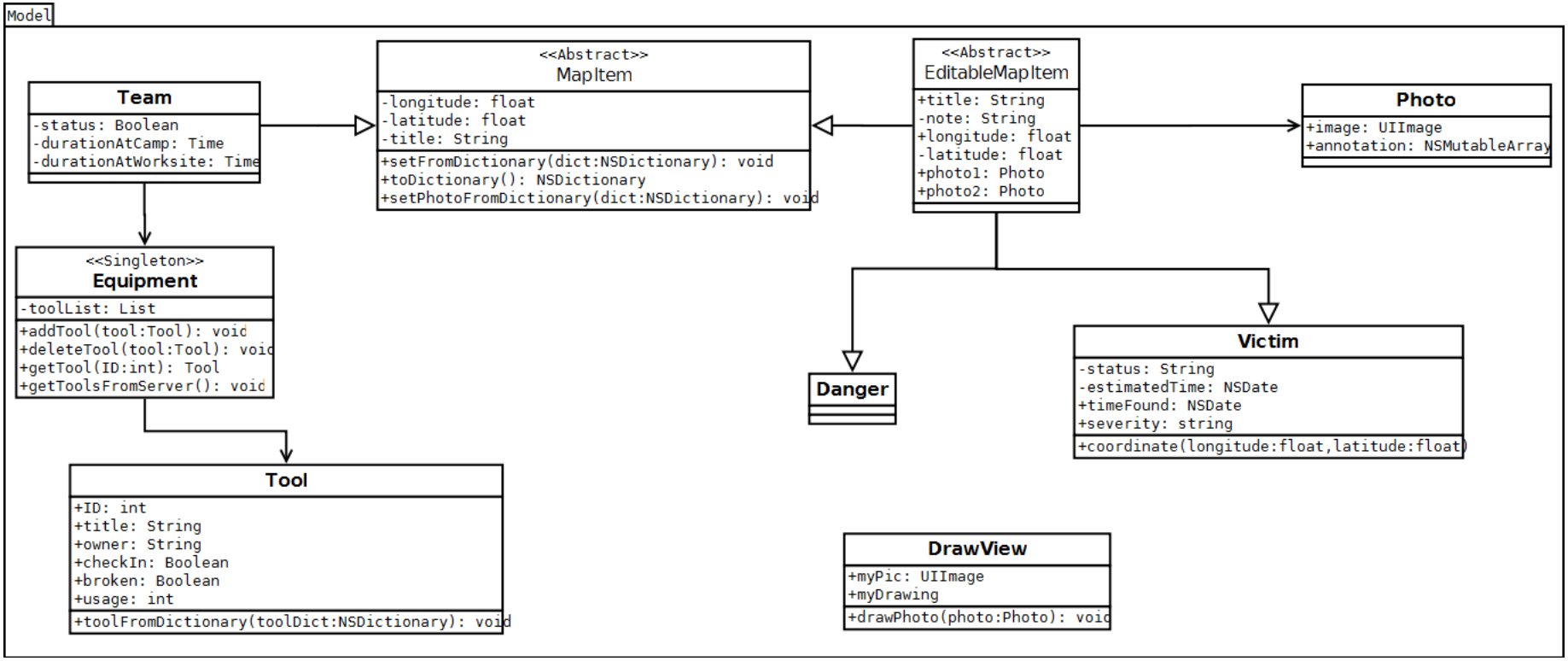

Figure 3.1: The Model.

2. **RFID Model** This model bellow is for the RFID Application

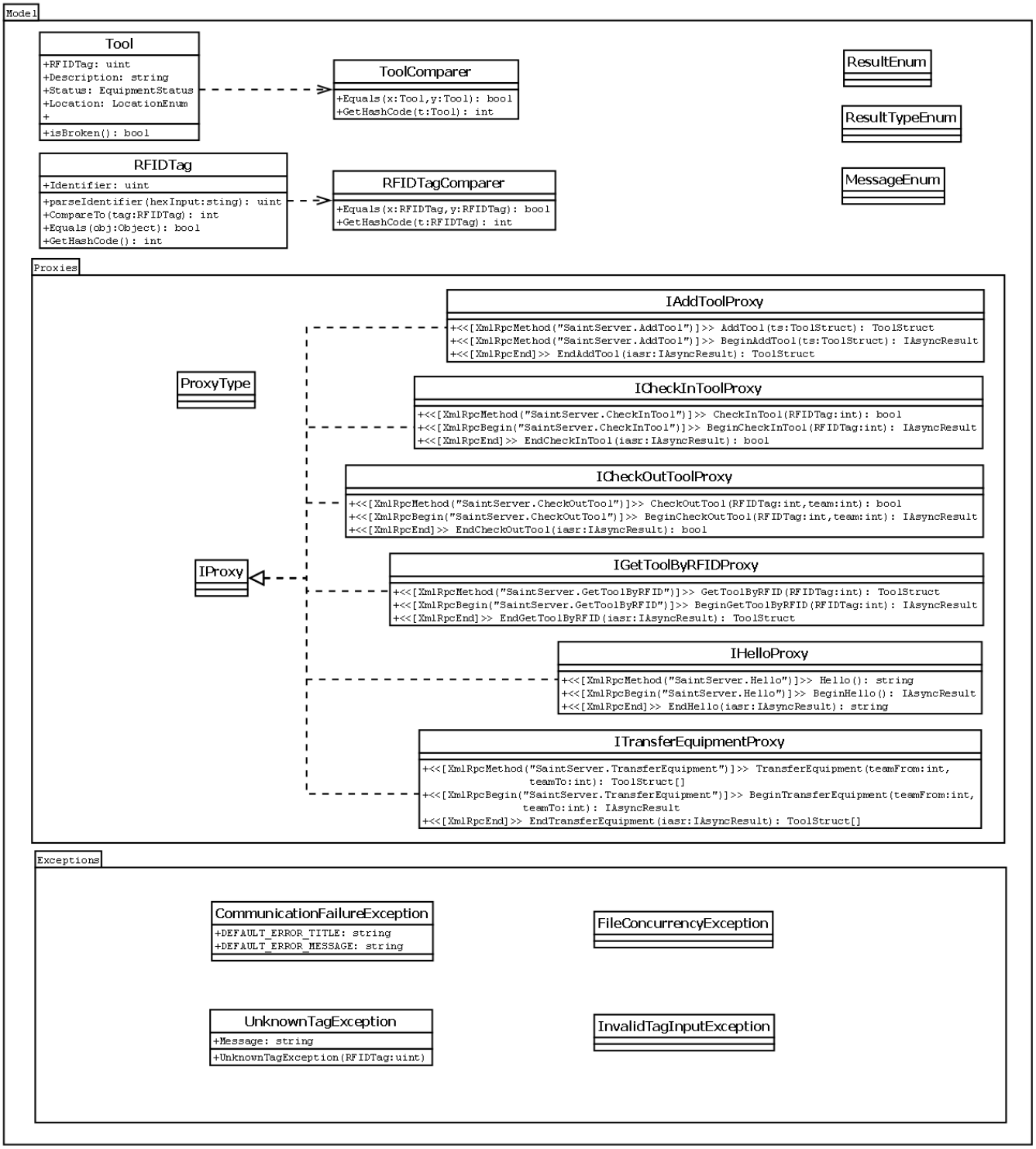

Figure 3.2: The Model.

3.5.2 Controllers

3. **iPad Controller** The Controller package of the iPad application contains all classes that are used to respond to the user actions, by providing methods to perform the required operations.

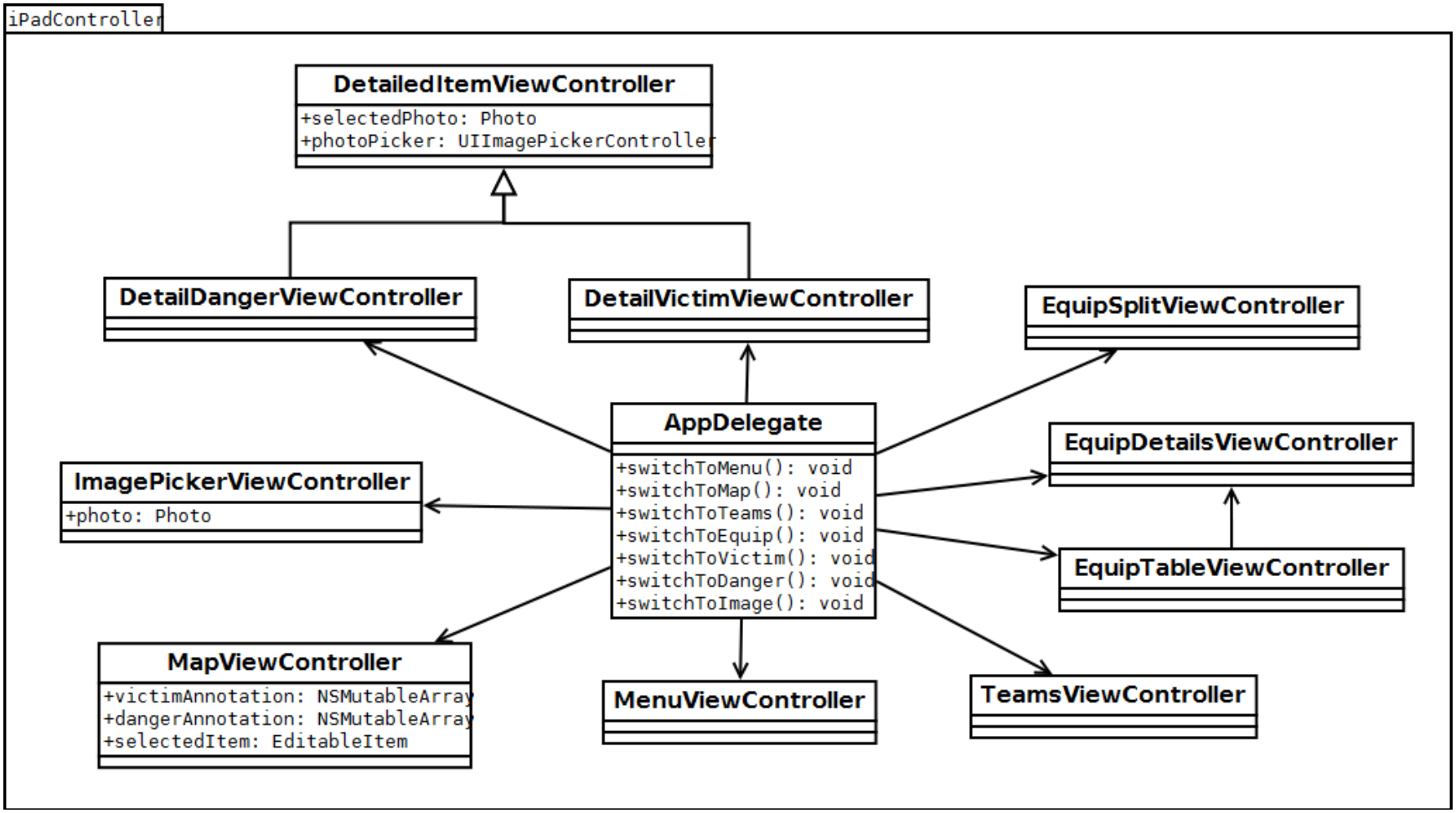

Figure 3.3: The Controller for the iPad application.

4. **iPhone Controller** The Controller package of the iPhone application contains all classes that are used to respond to the user actions, by providing methods to perform the required operations.

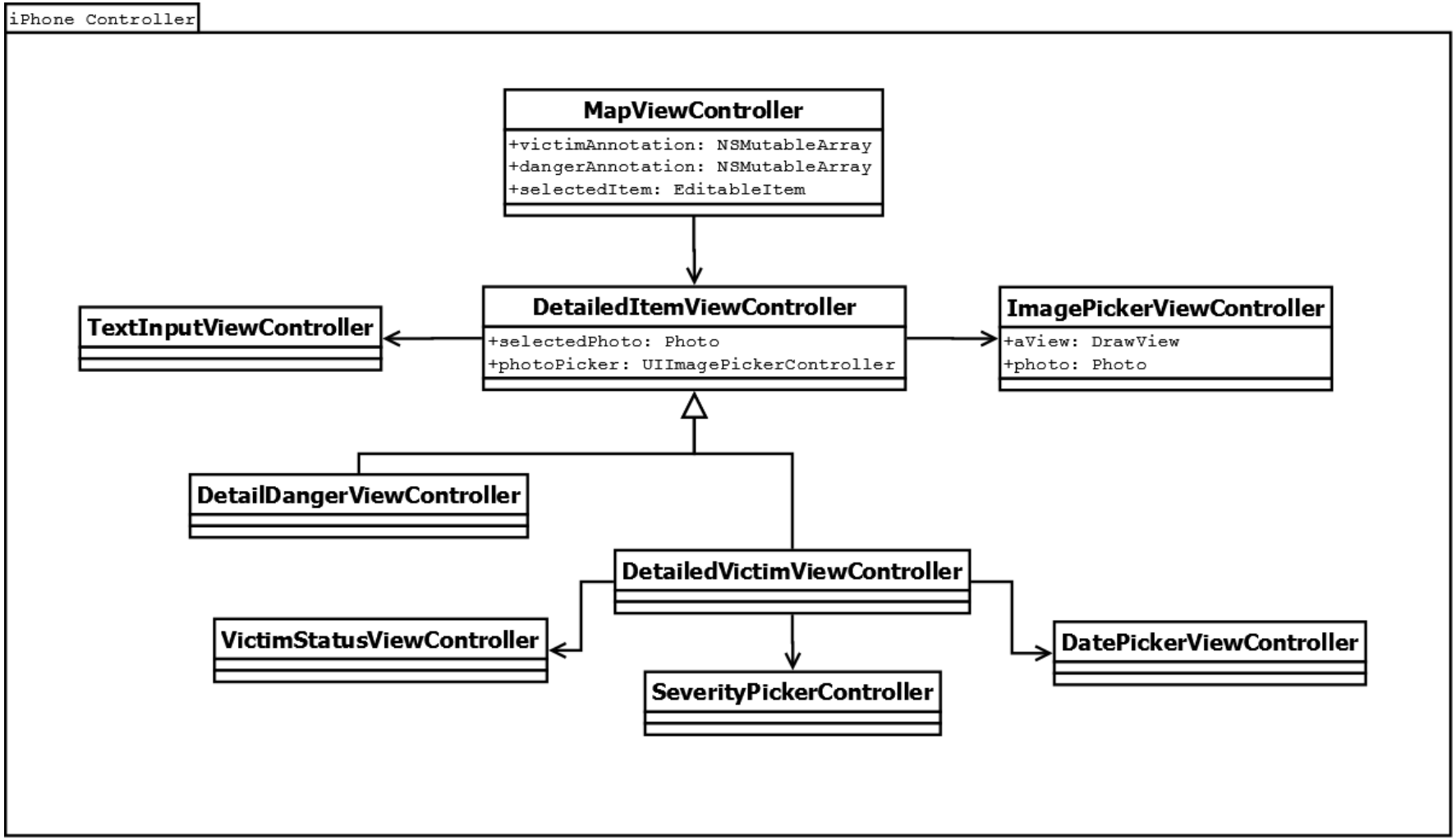

Figure 3.4: The Controller for the iPhone application.

5. **RFID Controller** The Controller package of the RFID application contains all classes that are used to respond to the user actions, by providing methods to perform the required operations. These operations also include communication with the RFID-reader, either through a library or by communicating with a separate program.

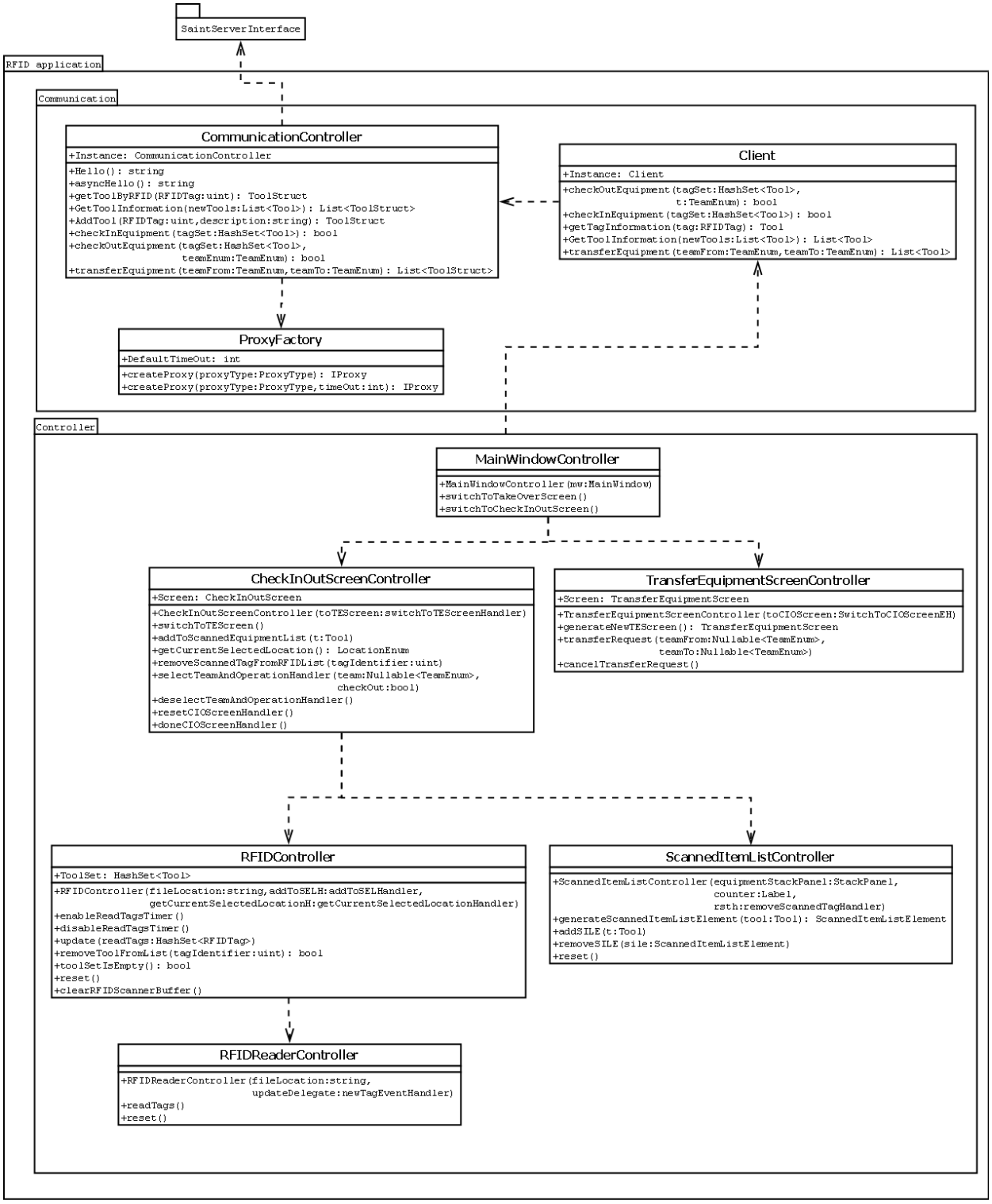

Figure 3.5: The Controller for the RFID application.

3.5.3 View

We have not included a view package diagram for the iPhone or iPad, because the views are represented as xib files (interface files for apple) and there is no interaction between them.

6. **RFID View** The View package of the RFID application contains all the classes that are seen by the user.

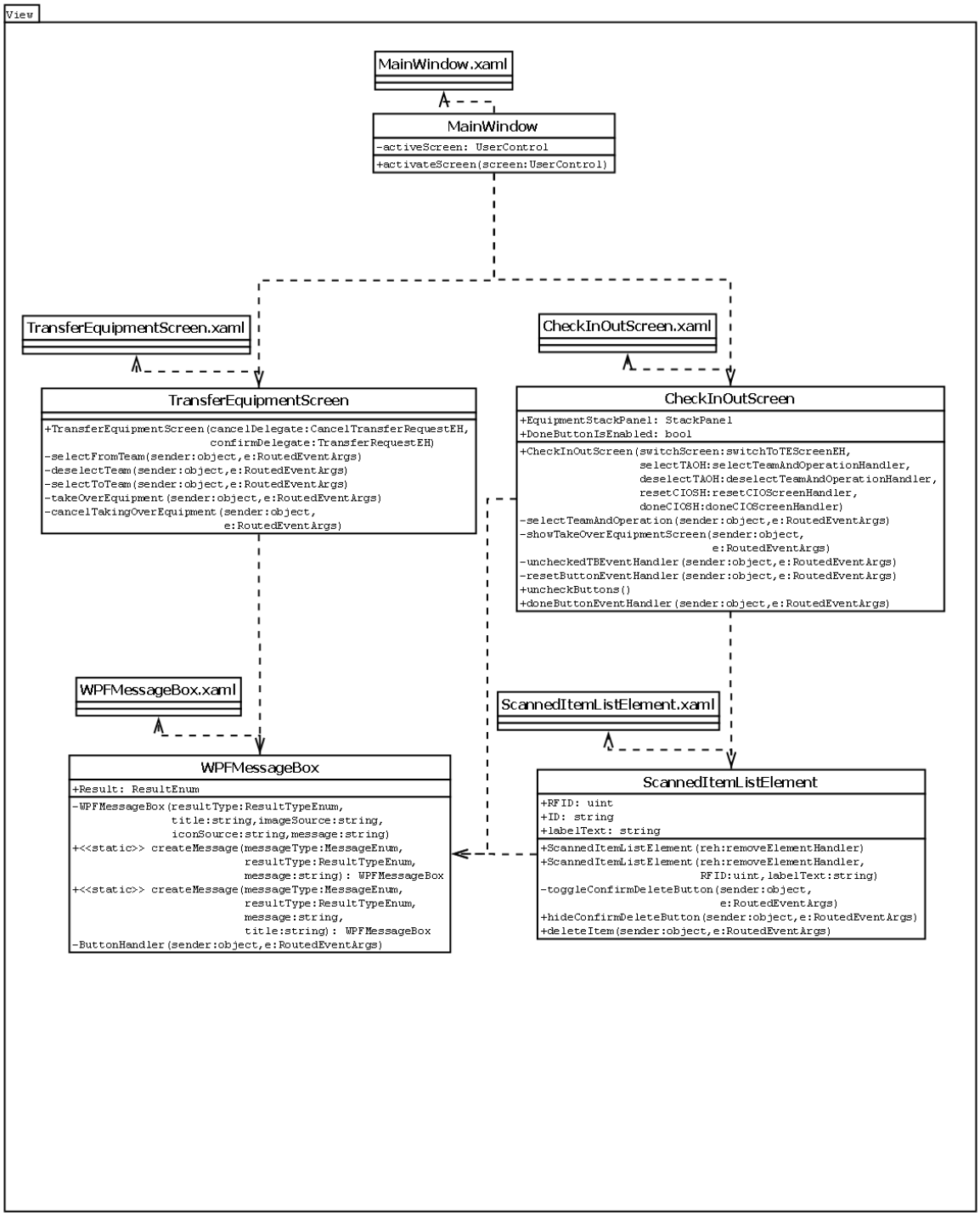

Figure 3.6: The Communication package.

3.5.4 Communication

The Communication package is used by the iPhone-, iPad-, and RFID-application to communicate with the server.

7. SAInT Server

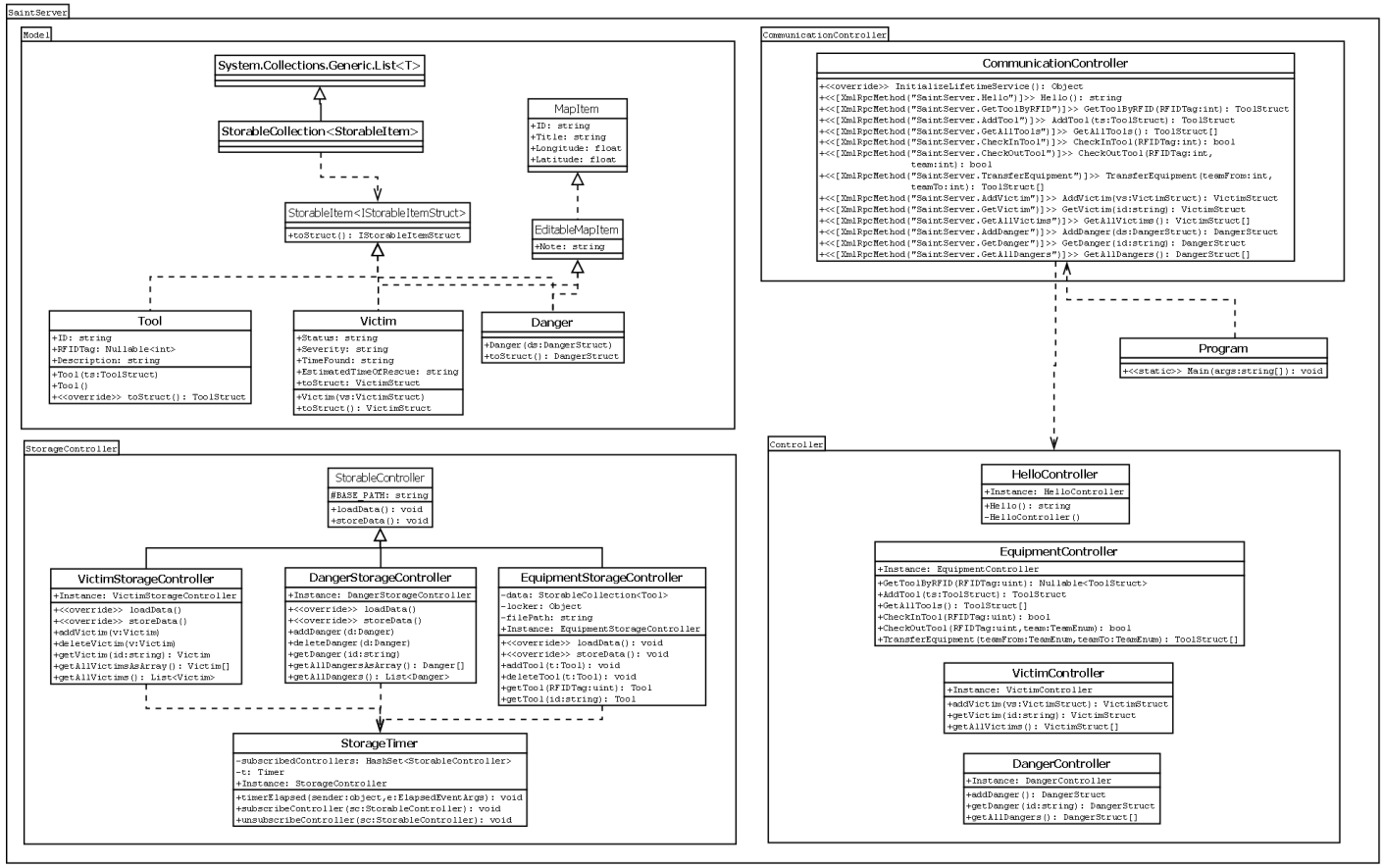

Figure 3.7: The Communication package.

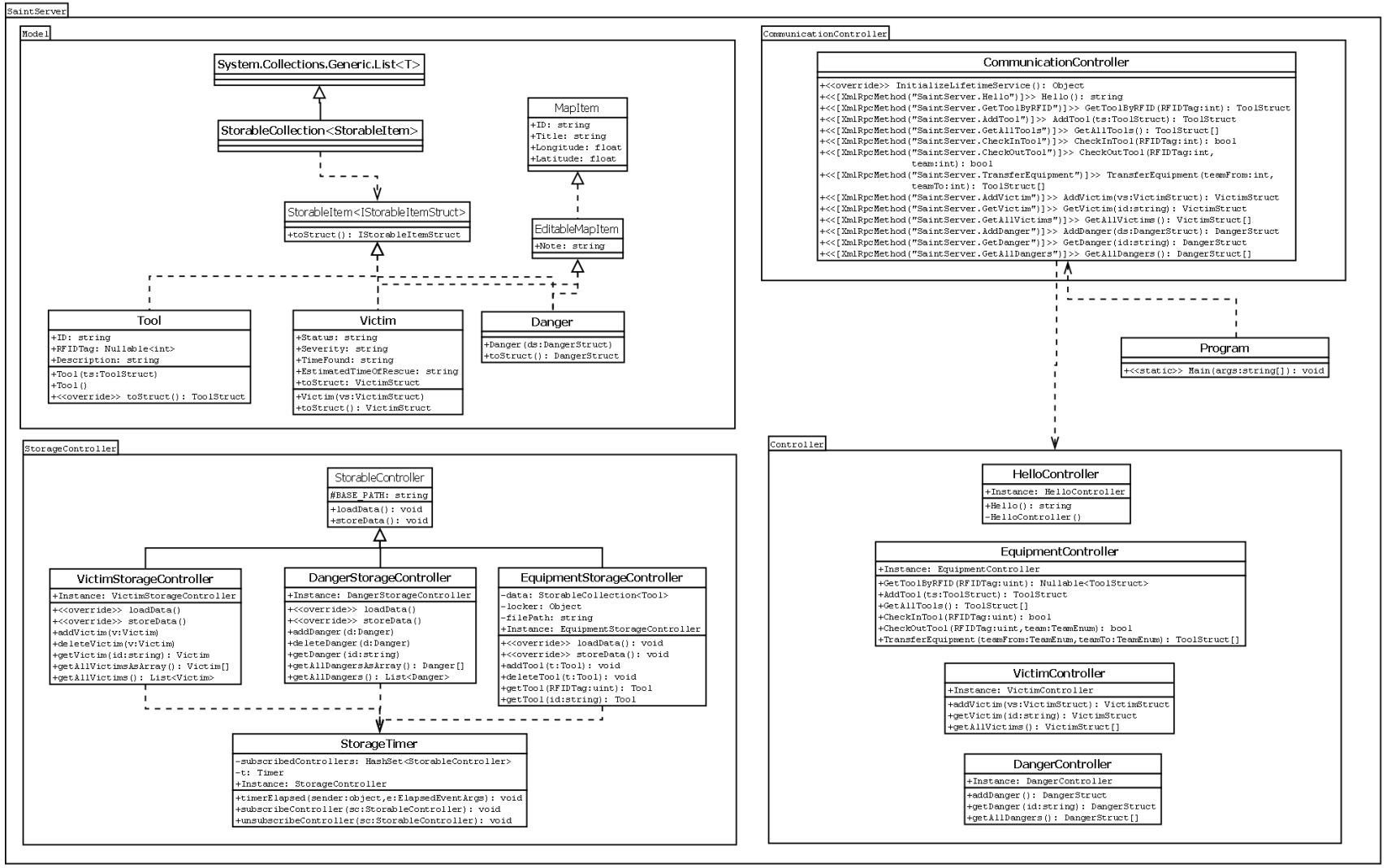

Figure 3.8: The Communication package.

9. **SAInT Server Interface**

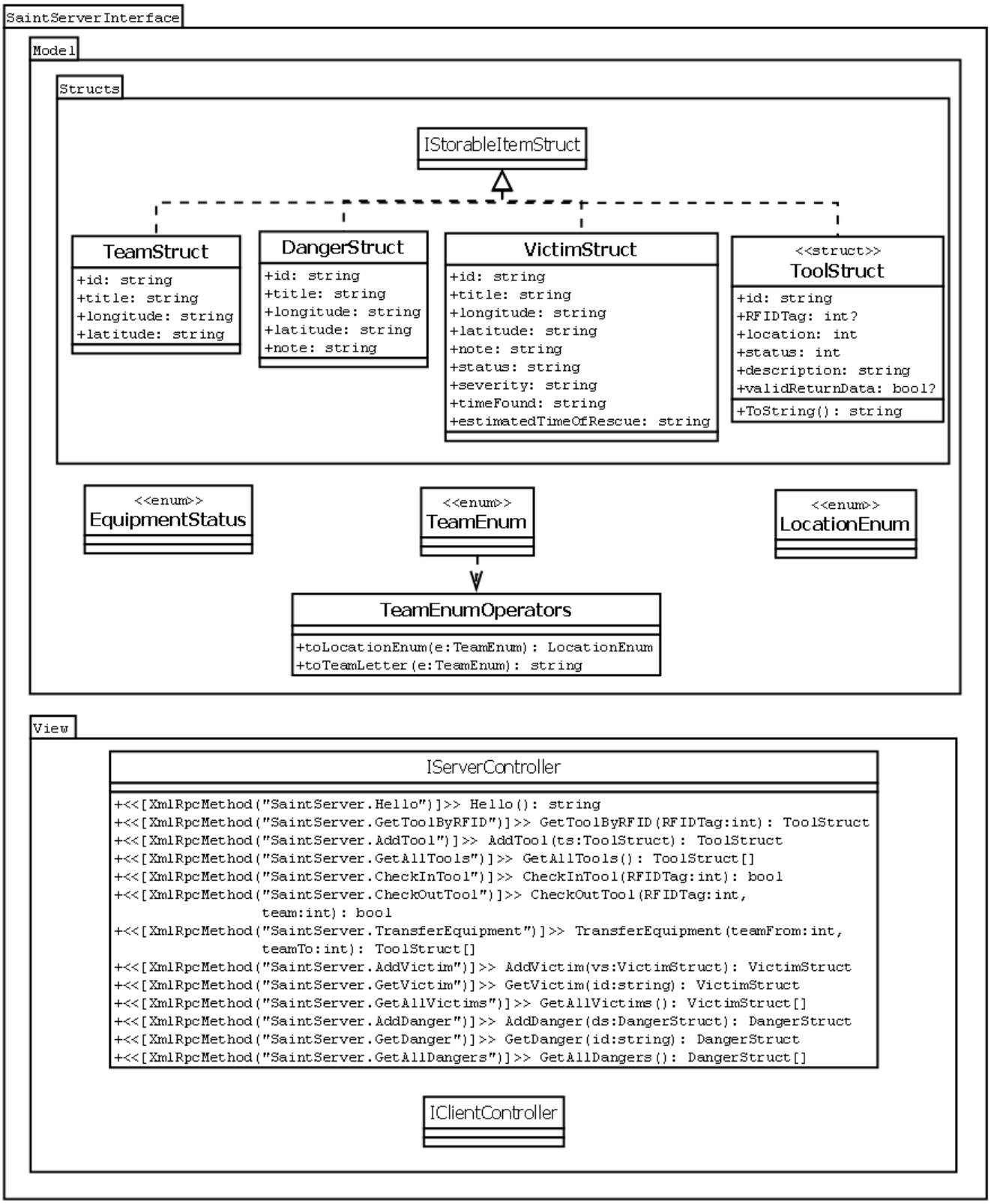

Figure 3.9: The Communication package.

3.6 Scenario

The following scenario shows how our system will influence the normal workings of an USAR organizations. This in comparison to the scenario in Appendix C section 1.1, it gives a nice before and after view of the operations of the USAR organization.

An earthquake measuring 7.4 on the Richter scale struck Turkey and USAR.nl has been called in for assistance. The entire organisation flew in with their equipment within the first 24 hours after the earthquake. Directly after landing at the airport, the operational commander has a briefing with the local authorities. Main topics of this briefing are the assignment of two work sites where USAR.nl will be working and the location where the base camp will be build. Furthermore, the commander gets a full update about all relevant circumstantial information, like possible threats of civilians plundering, embassy contacts, transport, supplies, weather information etc. The operational commander gets the last satellite map of the area on his iPad and identifies possible worksites, base camp location and possible dangers on it.

The commander of the support group is in charge of building the base camp. The entire USAR.nl group unloads the plane and starts building the tents for the different groups, e.g. specific tents for the recue groups, a medical tent and staff tent that serves as command post. All the equipment is stored at a central location in the camp and registered using the RFID scanner.

The work sites are located eight and fifteen kilometres from the base camp. The staff decides that team Alpha and Bravo are the first teams that start searching for survivors, so they call the commanders of the rescue teams either on their iPhones or on a satellite phone, and ask them to report in the staff tent. The latest information is already available on their iPhones as well. Both commanders are briefed in the command post using a surface table, showing a map of the area and all information that was provided by the local authorities. After the briefing the commanders pick the equipment they need, they check it out at the RFID scanner and they brief their teams and get on the transportation to the rescue site.

Team Alpha arrives at their work site and starts exploring the area. A few minutes later a partially collapsed office building is found, probably with live victims trapped inside. Another international USAR team already did a quick search of the building but did not find any live victims. They have spay painted a symbol on one of the walls to indicate this. However, locals insist that they have heard somebody, so the commander decides to do a scan with a search dog.

Team Bravo arrived and also starts to explore the second site, they quickly find a number of deceased victims and, after a few minutes, the first signs of life from under a collapsed house. It seems to be a family, consisting of four people. Through communication with one of the family members they find out that at least three of them should be alive. The commander adds these victims on his iPhone, along with detailed information such as photos with annotations..This data is automatically uploaded to the server.

After the thorough scan of team Alpha, they still did not find any signs of life. They spray-paint the wall that they didn't find anything and take a picture of this building. They want to proceed searching at the next building, but the local people, who still think their relatives are alive, start arguing. Because the situation is getting more and more hostile, team Alpha has to go back to the base camp.

The staff group in the base camp is tracking the progress of the operations. They manage the mission from the command post and therefore need to keep up to speed with what is happening out in the field. This is a difficult task, especially because they are at a distant location from the work sites. After 7 hours of work, team Bravo managed to rescue two of the family members.

The commander of the rescue team gives a situational report to the staff team to update them of the situation. They have been working for a long time and another team will have to take over. The staff decides to send team Charlie. The commander is briefed in the command post using the surface table, decides to take additional equipment after contact with the commander of team Bravo and checks these items out using the RFID application. Additionally, he also transfers all of team Bravo's equipment to his own team, again using the RFID tool and leaves for the site. Eight hours after team Bravo started working, team Charlie arrives at the work site. They need a good overview of the situation to take over command. Making use of the iPhone application, the rescue group commanders get together and discuss the most important events of the last 8 hours and future planned events. Important aspects like the structure of the collapsed building under which the victim is trapped are discussed extensively, and during the first actions of team Charlie, team Bravo stays to see if everything is going well. After half an hour team Bravo returns to the base camp.

Then, an aftershock occurs. This is a stressful situation for the staff group, members of USAR.NL might be in danger. Team Bravo is still on the way to the base camp and team Charlie is working in the collapsed building. Both teams are contacted and report that everyone is fine. Team Charlie needs to check the stability of the building they are working in. A staff member calls the structural specialist is for assistance.

At arrival the commander of team Bravo debriefs in the command post. All equipment team Bravo brought back from the work site is checked using the RFID application. A broken drill is taken to the support team.

Chapter 4

Implementation

In this chapter we will discuss our implementation process. In the beginning we worked on a basic user interface for our systems in preparations for a conference we had the opportunity to attend (which is discussed in the following section). After the conference we worked on the code behind the user interface adding functionality and extending the user interface where needed. Once each application for the iPhone, iPad and RFID were near completion was started building the server and the between the different applications.

4.1 Conference "The Web and Beyond"

Very early on in our implementation phase we had the opportunity to attend a conference "Web and Beyond" to present a demo of our product. "The Web and Beyond is the bi-annual daylong conference organised by Chi Nederland – and this year, co-organised by IOP-MMI – for the user experience and interaction design community." [1]

Due to the fact that we had only 6 days to produce a show able system (May 25 we started with implementation and the conference was on June 1) we had only enough time to make a basic user interface with mock ups and stubs and pop up alerts as substitute to functionality.

Even though we didn't have much time we thought it was important to produce something for the conference, since we are following the Situated Cognitive Engineering method. The prototype that we made for the conference would go through a iteration process where it is reviewed, we receive feedback and based on that refine our system.

During the demo at the conference we received positive feedback about our interface.

4.2 iPhone Application

Before we started this project a prototype of the iPhone applications was mad (GIS). The intentions was to build on the already excising program, however due to the fact that there was no documentation for the program, the code wasn't always clear. We did re use a lot of the ideas made with GIS, (the user interface was largely based on the GIS application). The application we made (SAInT) improves on the existing one by using updated components, for example SAInT we used the MKMapKit which was included with the iPhone OS 3 while GIS used an old version of an open source mapkit. SAInT also allows user to make and edit annotations on photos by drawing with ones fingers. SAInT has been made to be easily expandable and maintainable.

is also more maintainable, making it easier to expand.

beter te onderhouden werd gebruikt van oudere elementen photo annotatie

4.2.1 Model

The implementation of the model for the iPhone is slightly different than what was previously thought out to be. For example the GPSLocation class was removed from the model because it contained only coordinates and no methods. This created unnecessary complexity to the model if we created a new class so we placed the GPS coordinates in EditableMapItem.

There is also no map class, this is because we are making use of the MKMapKit which is a built in map package for the iPhone and iPad.

There has been major changes made to how the photo annotations are made. Due to the lack if time and the complexity of the design we opted for a simpler way of annotating photos. In the original design we decided that the user could place symbols and only arrows and circles. We believed that this would be easier to use than allowing the user to draw everything. How ever after some research we came to the conclusion that drawing the symbols was easier for the user and easier to implement. So instead of having a symbol, circle and arrow class there is only a PhotoAnnotation class with the original image and an array with annotations made.

4.2.2 Controller

Due to our inexperience with programming for the iPhone we had a basic idea how our control class package would look like. However this was not completely the case, we had to alter our controller class package design by adding extra controller classes to our design which we didn't account for. For example, every view there needed to be a View Controller. Also we needed to add extra controllers for the severity and date picker.

4.2.3 View

We tried to base our user interface as much as we could with the storyboards (Appendix C) and for the majority there was very little changed. Some of the positions of the buttons have changed and for the severity and date picker some labels were added to clarify to the user what he/she is picking.

Creating interfaces (views) is very simple and easy for the iPhone, because with the iPhone SDK (Software Development Kit) is an Interface Builder. The interface builder is a very handy tool allowing us to quickly produce interface, doing certain things automatically. However there were times that to much stuff were done automatically. As a programmer we were used to seeing things happen due to code and there were times unexpected things happened and it took a while to figure out why (happened through the interface builder). It also took a little while to get the hang of how the interface builder makes connection for example, using the navigation tool.

4.3 iPad Application

Our inexperience with the objective C language was a disadvantage when designing our the iPad application. During the implementation process, several changes to the design have been made to better suit the language and the model view controller style. Overall though, no major changes were required in order to make a reliable and maintainable product. This section will describe the design choices made during the implementation phase with regard to the iPad application.

4.3.1 Model

The model implemented in the iPad application is slightly different from the initial design in several aspects. One new class has been added, the TeamContainer class. This is a Singleton class to save the four Teams. These teams needed a place to be stored when retrieved from the server. Instead of using a controller class to take care of this, the TeamContainer class preserves the model view controller structure.

Every individual Team (four in total) is now a subclass of MapItem. This change has been made, in order to show the team as annotations on the map. Without inheritance, an extra annotation object would have to be created for each team. Allowing the Team class to inherit form MapItem also allows the Team to inherit several properties, including a GPS location and a title.

The GPSLocation class has been removed from the model. Instead, every MapItem now has a two additional variables, a longitude and a latitude. This change has been made to reduce the complexity of the model. The GPSLocation class would merely require extra calls to be made to this class to retrieve the coordinates. It would not need to contain any additional methods or perform any special operations.

Apart from the changes mentioned above, both the Client and the Equipment classes have been changed as well. There is no need to ever create more than one instance of these classes. For that reason, both classes have been converted to Singleton classes.

Finally, the Map class from our initial model has also been removed. This has been replaced by using the iOS (previously known as iPhone OS) MapKit. The classes in this kit contain everything we previously anticipated to use in our Map class.

4.3.2 View

The view was created using Apple's interface builder, which has a couple of advantages. First of all, it is very easy to keep the style of an interface similar to Apple's style. All buttons, for example, have the default layout as seen in many other applications. Another advantage of interface builder is the method of creating action handlers for the buttons. This can be done by simply dragging and a line from a button to a class method. The prototype was completed just in time to show our applications at the conference.

Due to certain limitations of the interface builder and some feedback from other people, several changes have been made to our original storyboards. Mainly, several buttons have changed position. A navigation bar, at the top of the screen, can only contain a title and two buttons. If you would want to have more buttons, you would need to use a toolbar and you can no longer show a title. Because this title is very useful for a user, several buttons have been moved to different positions.

The main change, however, is in the team statistics table. Instead of a table similar to the one in figure 10 of the requirements analysis document, we have used a table in the style of all other iPhone and iPad tables. This new table consists of four different sections, one for each team, and constitutes to a clearer and nicer look.

The ability to switch between the map view and team statistics view directly has been removed as well. This has been done to avoid confusion, because either one of the views could be considered as a subview of the other, which is not the case. This change allows for a repositioning of the other buttons in the map view, improving the overall look of this screen.

Finally, a new option has been added to switch the type of the map between a satellite image, a road map and a hybrid view. This could allow a rescue worker to view whichever type of map would be most appropriate in his current situation.

4.3.3 Controller

The controller classes also differ from our original design. Due to our inexperience with iPad programming, we did not anticipate the need for some of the view controllers. For example, to use a split screen view, like the one we currently use for the equipment screen, three separate view controllers are needed. Additionally, a special RootViewController, like many iPhone and iPad applications have, would not be appropriate for our iPad application. It would merely be responsible for showing a single subview, something the AppDelegate could do as well.

4.4 RFID Application

In this section, we describe the design choices made while designing the RFID application.

4.4.1 System requirements

In addition to the system requirements described in the Requirements Analysis Document (Appendix C), the RFID application has the following system requirements:

The RFID Reader required a separate application, since the application needs to communicate with the software provided by the manufacturer of the RFID reader, which only runs on Windows. The computer also needs one free USB-port to connect the RFID-reader.

This software requires .Net 3.5 Framework or higher to be installed, since the application was developed in C#. For compatibility reasons, the application would have to be developed in Objective-C, the programming language the iPad and iPhone applications are developed in. However, since the Windows support for Objective-C is very limited, and the Surface Table application is programmed in the C# programming language, the RFID application was developed in C#.

4.4.2 Structure of the application

The application is set up using the Model-View-Controller (MVC) pattern. In addition to the Model-package containing the data classes, the Controller-package containing the operation logic, and the View-package containing the interface classes, we have included a Communication-package containing the logic needed to connect to the server and transfer data. During the implementation, the general structure has not changed.

4.4.3 Interface

The interface of the RFID application is created in XAML and C#-code using Visual Studio as a User Interface (UI) editor. The appearance of the application was styled using XAML, and all actions were handled using the C#-code. This allowed for separate development of the GUI for the conference, and the code-behind afterwards.

The UI was developed for use with a touch screen. No other input methods than the touch screen and the RFID-reader are required. The UI contains large buttons, which makes selecting the relevant option on the screen easier and more accurate than with the regular, smaller, buttons.

The UI is designed with simplicity and reliability in mind. The UI needs to be as simple as possible, requiring a minimum of user actions. With normal use, only three interactions with the UI are required to check any number of tools in or out: To select a team/option, to confirm that the scanned tools have to be checked in or out, and to dismiss the confirmation dialogue

that the tools were checked in/out successfully. The transfer of equipment from one team to another requires only five user actions.

One of the non-functional requirements that was formulated in the Requirements Analysis Document is, that the system must be reliable. The user interface was designed to minimize possible errors, by disabling options when those options are not allowed to be used in the context the interface is in. When an error occurs, the user is notified of this by a dialogue explaining the nature of the error, and a possible action the user can take. This way, the risk of erroneous information in the database is minimised.

4.4.4 Reading RFID-tags

In order to read the RFID-tags, the RFID application must interface with the RFID-reader. Since there are many types of RFID-readers from different manufacturers, we decided to contact the RFID research group of the TU Delft CiTG department. There, an RFID reader and several passive RFID tags were available for testing purposes. For this BSc-project, we decided to design the application for use with the Deister RFID reader present at the RFID research lab for convenience and availability of the reader. However, the RFID application is designed to be easily adaptable to work with other types of RFID readers and -software.

The disadvantage of using the Deister RFID reader is that there is no interface between the reader software "RDemo" and other programs. Therefore, we used the auto-logging functionality of the RDemo-software, which writes the identification number of the read tags to a log file. This log file is then monitored by the RFID application, which processes the RFID-tags, and presents the items matching with the RFID tags in the user interface.

A few issues still exist when using the RFID-tags and RFID reader. The reader is not adequately cooled, and gets very hot when operating for prolonged periods of time. In hot circumstances, for example in Dubai where the USAR.NL has done their last training, the reader may fail. The tags also need to be attached more than 2 centimetres away from metal surfaces, since the interference of the metal makes it impossible to read the tags. The tags are also very vulnerable, since these are "sticker-type" tags. Using these tags and this reader in the final version of this system will cause reliability issues. We recommend using tags with a plastic casing, which are resistant to nearby metal, and a tag reader with better cooling.

4.4.5 Communication with the Server

XML-RPC is used for the communication with the server. With this protocol, HTTP-messages are created which represent function calls to the server. The answer is then returned to the client, which can process this information. The messages are proxies that contain the method signature of the method to be called on the server, which are instantiated using a ProxyFactory-method within the Communication package. The object that can be sent to and received from the server are defined in a separate class library of the server called "SaintServerInterface".

4.4.6 Implementation process

Implementation started the 25th of May. The main priority was then, to create the interface of the RFID-application to present at the conference "The Web and Beyond 2010" before the first of June. The implementation of the interface went well, and we were able to present a working interface at the conference.

After the conference, work on the RFIDReaderController of the RFID application started, since the top priority for implementation was the checking in and out of equipment. In order to

read and process the files in the RFID application, some example log files were needed to test the RFIDReaderController.

Then, the parser of the file was implemented, including a timer that calls the parser every 500 milliseconds to parse the file. Initially, this did not work, as our program was locking the file to be read from, which caused the Deister software to crash. This problem was solved by not locking the log file, but this approach required the parser to be very robust, since at any time, new tag data can be written to the file, which could cause concurrency problems if these are not prevented.

With the RFID application now capable of scanning tags, the parsed tags would have to be displayed on the graphical user interface (GUI). This meant that, for each scanned tag, information would have to be retrieved from the database, and a new list item displaying the description of the tool, the tag ID itself, and a button to delete the element from the list. This caused some threading issues, since the parser of the log file was called by a timer, which caused the parser to run in a separate thread. This meant, that all calls to the GUI to process the new tag data would need to be synchronised with the GUI thread. This problem was solved by using a different timer, which used the dispatcher of the GUI to call the parser instead of calling the parser based on the system timer. After a test with the saved log files, the regular scanning of tags worked.

The following days, a first version of the code-behind for the remaining part of the GUI was created, for example the button handlers, and temporary storage and manipulation of tool data, except for the communication with the server. No major problems came up while implementing this part of the RFID application.

After re-implementing the server, the communication with the new server was set up. Most communication with the server is done synchronously, since only one request is sent to the server. When multiple requests are sent to the server at once, this is done asynchronously, to optimise performance by issuing all requests at the same time. This is done to minimise network latency, which can be high when accessing the server through satellite connections or through the mobile telephone network.

Lastly, we improved the feedback of the application, by designing message boxes in the same style as the application itself, which are used to display urgent information or confirmation of actions.

4.4.7 Testing

All functionality has been tested by executing use cases manually, including variants of these use cases, to check if the program exhibits the expected behaviour. Logging functionality and the debugging functionality of Visual Studio were used to observe the state of some parts of the application while running.

Some parts of the RFID application, such as the RFID log file parser, were tested using unit tests. A class invariant was used in the RFIDController class to ensure the RFID controller could not enter an illegal state.

These observation methods, manual tests, unit tests, and class invariants have proven useful in preventing and finding faults in the code, especially when the concurrency issues with the Deister RDemo program needed to be fixed. Some faults would otherwise not have been found.

4.5 Server Application

In this section, we describe the design choices made while designing “SaintServer”, the server application that is used by all other programs to store and retrieve data.

4.5.1 System requirements

In addition to the system requirements described in the Requirements Analysis Document (Appendix C), the SaintServer requires .Net 3.5 Framework or higher to be installed, since the application was developed in C#.

4.5.2 Structure of the application

The application is set up using the Model-View-Controller (MVC) pattern. In addition to the Model-package containing the data classes, the Controller-package containing the operation logic, and the View-package containing the interface classes for connecting with clients, we have included a StorageController-package containing the logic needed to store and retrieve data from the storage files. The structure was designed with extendibility in mind, by creating an application with several layers of abstraction. The view is responsible for receiving the message, and calling the responsible controller. This controller contains the logic for the operation, and makes mutations to the collection of objects held by the associated storagecontroller. The associated storagecontroller holds all objects of a single type, and is responsible for the storage of those objects.

4.5.3 Persistent storage

The persistent storage of data is done by saving the collections of data (victims, equipment, etc.) to an XML-file every minute. A separate timer is set up, which calls all StorageControllers to save their data to the XML-file. When the server stops working for any reason, the backup of the data is at most 1 minute old, so data loss is kept to a minimum.

Another option we have considered, is using a MySQL-server running an SQL-database. Since we had not planned on re-implementing the server, as we were told that a server existed, we did not have time to design and implement an SQL-database.

4.5.4 Communication with the Clients

XML-RPC is used for the communication with the clients. The server receives messages over HTTP, which represent function calls to methods on the server. When a message is received, the method to be called is determined and that method is dispatched to the correct controller. This controller then processes the request and returns the result. This result is then sent back to the client.

4.5.5 Implementation process

The original server was not designed to contain any other data than victims. As the structure of the original server was too tightly-coupled, we decided to re-implement the server, since this would be easier than to adapt the original server to our needs. This was a minor setback in our schedule.

We first implemented the general structure of the server: The communication and the storage part. The first methods could be implemented and tested after two days, after which the server was ready to extend as needed.

Then, the communication packages of the iPad, iPhone, and RFID application could be implemented to send and receive messages from the server. The SaintServer could then be extended to process the requests and store the new data

4.5.6 Testing

The original structure was tested using an "Exerciser", which mimics a client application and calls methods on the server. The results of those methods were then shown in the console, to be checked manually. Other methods were tested by printing messages to the console of the server and the exerciser.

4.5.7 iPhone and iPad Client

Due to the new server, we have also decided to create the client on the iPhone and iPad from scratch. This allowed us to make several changes compared to the client used by the old iPhone application. First of all, the Client class has been converted to a Singleton. This change has been made, because there would not be any need for more than one client at any one time. Secondly, this allowed us to make a large amount of minor changes, resulting in a cleaner and more maintainable class overall.

Chapter 5

Conclusion

The goal of this Bachelor's project was to improve information exchange within the USAR.NL-organisation by focussing on providing methods for information input, automatically gathering information, and information presentation. The system we have created, the "Search and rescue Awareness Information Tool" (SAInT), improves information exchange on all three areas.

To determine the information needs of the different teams, an extensive analysis was conducted, by using the sCE method developed in the "Collaboration at a distance" project of TNO and the TU Delft. The first step in this method consists of investigating organisational demands, human factors knowledge, and envisioned technology. Then, requirements were specified, which included the types of information to be collected by and presented to the different teams in order to improve collaboration at a distance and situational awareness of the teams.

Based on the sCE-analysis, we designed the three applications that support the USAR.NL team, and form SAInT:

First, we designed an iPhone application that is used by the commanders of search and rescue teams. The commanders use this application at the work site to input information about victims the team has found, and dangers this team and other teams must be aware of. Since the commanders have little time when rescuing a victim, the application only contains the relevant functionality, which is easily accessible. Collaboration is improved by supporting the sharing of information within the USAR organisation, and by improving situational awareness of the USAR team commanders by providing an overview of the situation at a work site.

Second, we designed an iPad application that is used by members of the command team and the technical workers. The commanders use this application to obtain an up-to-date, comprehensive overview of the whole USAR operation. In order to communicate effectively with local government and UN representatives, and the coordinating USAR.NL-team in the Netherlands, the commanders need to quickly obtain situational awareness at any time during the mission. The application provides the members of the command team access to large amounts of relevant information, for example a map of the region, including ongoing rescue operations, and performance statistics on the teams. The technical workers use the iPad application to view information about all equipment that has been brought to the area the disaster has occurred. This way, the technical worker can view and register what equipment is broken and needs repair, so the search and rescue teams do not bring items to a work site that are broken.

Third, we designed an RFID application that is used by members of the search and rescue teams. The search and rescue team members use this application to check out the equipment they want to bring to a work site, and check the equipment back in again when they return to the base camp. RFID technology is used to identify the equipment quickly and easily. This way, information about which equipment the team has taken with them, and to gather information about the use of equipment. This information can then be used by other teams to decide which

equipment they have to take with them to the work site when relieving another team.

All three applications communicate with a central server, which enables the applications to store, retrieve and share information. It will also be used to share information with the Surface Table application that is being developed by Sander Ton, a master student that also works on this project.

SAInT will improve collaboration at a distance, by providing the required information at the right time. This will allow the USAR.NL team to operate more efficiently, and rescue more victims during the mission to a disaster area. All three applications form a single solution for improving collaboration at a distance in the USAR.NL team to achieve this goal.

Chapter 6

Recommendations

Although our work on this project has finished, a lot of work can still be done to improve the current system and the collaboration between different members of the USAR.nl organisation and to decrease their dependence on older methods or less optimal systems. In this chapter, we will focus both on the functionality we have not been able to implement due to time limitations as well as on new functionality we have considered during the implementation phase. Section 1 will describe the recommendations according to our requirements, while section 2 describes new functionality we envisioned during the implementation phase. Finally, section 3 lists any known bugs.

6.1 MoSCoW

Appendix E contains the test and implementation plan. Chapter one of this document contains the requirements ordered by priority according to the MoSCoW system. During the last months, all 'must haves' and a large part of the 'should haves' have been implemented. However, a lot of improvements can still be made. One of the most useful additions would be added support for sound recordings when adding victims and dangers on the iPhone. This could replace the need to type a note explaining the condition of a victim, which can be a very time consuming task. Instead, the rescue worker simply presses a button to start recording, leaves a message and finally presses another button to stop recording.

A useful addition to the iPad application would be to improve the list of equipment by implementing various filter and sort functions. A member of the support team, for example, could filter the list to only show broken items. A member of a search and rescue team could filter it to see all available equipment or search for a specific tool.

Both aforementioned improvements would be relatively easy to implement. With additional time however, some more extensive improvements could be made as well. One of these would be to provide the command group of situational reports on the iPad. Currently, the staff group already writes these reports. With some minor changes, these could simply be sent to the server, making them available to the iPad application.

A timeline, similar to the one available to the staff group on the Microsoft surface table, could also benefit the command group. Combined with the overview of the teams and victims on the map and the team statistics already available, this would provide increased awareness of the actions of each individual search and rescue team.

The support group could benefit from a couple of improvement to the iPad application as well. With the ability to update the status of equipment and add maintenance reports to items, the support group would not need any other application when repairing equipment. This will enable them to move around the base camp more freely, carrying merely the portable iPad.

At the moment, it is impossible to attach an RFID-scanner to an iPad. This is, however, something that could be implemented in the future. While it is possible to attach different types of devices to the iPad using either the dock connector or bluetooth, no company has done so with RFID-scanners so far. And option like this could allow the rescue workers to attach their maintenance reports to tools more easily, by simply scanning their tags. Adding new tools too the database would also be significantly easier.

The new check-in/check-out system for equipment will increase the time a search and rescue team needs to get ready to leave the base camp for a mission. While the scanning of equipment is not a very time consuming task, it could still be annoying to do when a team wants to leave quickly. Automatically checking tools in and out, by, for example, passing though a gate, would require less work from each team compared to manually scanning every item. This does, however, have several complications, mainly with regard to reliability of the system. If some items are not registered whenever a team passes through the gate, the data can not be fully trusted and will therefore become useless.

At the moment, members of a search and rescue team paint symbols and text on walls, indicating where a victim is located, whether he or she has already been rescued or not and several other statistics. By registering these actions automatically, data about victims could be gathered without the need for any user input. This will provide the other teams and the staff group with information, while it does not take the rescue worker painting the symbols any additional time. This could be implemented by attaching an iPhone to the spray can. The accelerometer in the phone could be used to detect the movement of the device, and thereby detect the movement of the spray can. While a similar device could also be used for this, Apple's phone would be best, since it is already being used for the SAInT application.

Finally, our systems still needs to be fully integrated with the service table application. Currently, all information provided by our applications is being transferred to the server, but this is not yet available to the staff group. The appropriate XMLRPC methods should be implemented on the surface table in order to retrieve this. Additionally, our current system does not use the information provided by the service table. Making this information available on the iPad could improve the awareness of the command group.

6.2 New Functionality

During the implementation phase, some new ideas for future development arose. One of these ideas was to implement a message system on the iPad and surface table to allow the staff and command groups to communicate with each other. A system like this would be similar to a mailing client, but could be integrated into the SAInT application. This would remove the need for internet access or phone calls to get in touch with the commander or staff group. Instead these messages will be sent using the wireless network already in use for the other SAInT systems.

Additionally, the iPhone application could also use a reduced version of this message system. In this version, the commander of a search and rescue group would probably not be able to send messages, but he would be able to view short, sms-like, texts sent by the staff group. These texts could be a simple request to contact them whenever the rescue worker would have the time to do so.

Another improvement could be made to the RFID-application. Currently, it is only possible to view equipment and their location on the iPad. A useful addition would be to make this information available to someone checking equipment in and out. When replacing another team, this would remove the need to contact the team to be replaced or to view the equipment on an iPad. Instead, rescue workers can see which tools another team is currently using and decide

which tools they would need to bring to the work site.

6.3 Known Bugs

For the iPhone there is a small bug, for the very first time for each item (victim or danger) a photo is made the photo the photo annotation view shows the photo with a white bar at the bottom. However if one then tries to make an annotation the picture will resize to the right format (without the white bar at the bottom). Then after for rest of the time when a photo is made for that praticular item it will show the picture in the right format. Even if you delete all the photos and then make a new photo. We suspect that the bug has to do with the annotation (because the image itself draws correctly in the image without annotation mode).

Chapter 7

Reflection

In the course of this Bachelors project, we have designed and implemented a support system for the USAR.NL team. In this chapter, we reflect on the process and the method we have used to derive the requirements for this project.

7.1 Process of Design and Implementation

At the start of our Bachelors project, we had already read some papers on the USAR-domain and user-centered design, which were given to us by our project manager Marc Grootjen. In the first weeks of the project, we conducted an analysis of the domain using the sCE-method. Since we did not know enough of the USAR-domain to perform this analysis, we could discuss these problems with our project manager Marc Grootjen and TU coordinator Tjerk de Greef, who were also our domain experts, during the weekly meetings we had with them.

The design process itself took relatively long, as we used a method of acquiring requirements which we were not familiar with. During this design stage, everyone on the project team worked together to derive the requirements, by cooperating in brainstorming sessions to discuss the possible design options, and making the design decisions. These decisions were then worked out in the requirements analysis document individually, with each member of the team working mostly on the requirements for their part of the project, as assigned in the project description.

After the design stage, we started implementing the different applications, each on the part that was assigned to each of us in the project description. The cooperation within the team went well, with each of the project members working on their own parts individually, keeping to the implementation schedule. On the parts that would have to be shared between the application, such as the map-part of the iPhone and iPad application, or the implementation of the server, implementation effort was combined to minimise the amount of time spent waiting on the shared part of the application.

In conclusion, we are content with the cooperation within the team, and with the project manager and TU coordinator. We have all put a lot of effort in this project, working full-time, and sometimes even in the weekends, to create this application.

7.2 Situated Cognitive Engineering Method

The situated cognitive engineering (sCE) method was developed in the “Collaboration at a distance”-project at TNO and the TU Delft. The sCE-method primarily focusses on the users. In the first step of this method: deriving the operational demands, human factors knowledge, and envisioned technology, the technology available today is not a determining factor. These

three areas offer different insights that allow for specifying the requirements. This part of the method is very different from the software developing methods we have learnt before. By first looking from the point of view of the user, we gain insight in the actual needs of the users, which is essential for developing user-friendly and effective applications.

Specifying the requirements, and making use cases to contextualise those requirements, is a key step in every software development method. The difference with this method is, that specifying requirements is a highly iterative process, in which the analysis conducted before serves as a knowledge base. This method helps to make design decisions explicit by referring to the analysis conducted in the first step. This improves traceability. First, the core functionality is determined, after which the requirements are formulated. Then, claims are made about those requirements, to make these requirements testable. This makes the process of validation easier.

The testing and refining steps we have conducted in this project primarily consisted of expert reviews. We have consulted our experts many times to refine our requirements, since we did not have enough time to test our program with the users. This may be done in a later stage at TNO, after this project is finished.

We were the first to use this method for developing an application, since the method is still being developed, so, naturally, we have encountered some problems while using this method. One suggestion for improvement, is differentiating between functional and non-functional requirements when specifying the requirements. This method focusses primarily on the functional requirements, which makes it difficult to specify design decisions made about the system as a whole. Non-functional requirements are not part of the design philosophy of the sCE-method, since these requirements are technology-centred and not user-centred, yet they are very important for the software design process that follows. Non-functional requirements provide demands that are testable when the implementation is finished.

Another suggestion is, to differentiate between a research-oriented project and a software development-oriented project. Some aspects of the sCE-method, such as defining positive and negative claims, are less useful for development-oriented projects, since the focus is very different. Research-oriented projects tend to focus on the future possibilities for a system. Formulating positive and negative claims then helps document potential pitfalls. In a development-oriented project, the focus lies on the aspects of the system that can be realised now or in the near future. Potential pitfalls are then already known to developers, so these claims are less useful.

In conclusion, the sCE-method is very useful for eliciting requirements from a domain from which little is known before starting the project. The method still needs some adjustments to make it more suitable for use in development projects. Also, the documentation of this method needs to be improved, since it raised some questions about the relation between the parts making up the “specifying”-step.

Appendix A

Project Description

The following pages contain the original project description, as provided at the start of this BSc-project.

Bachelor opdracht

Het verbeteren van de informatievoorziening binnen USAR.NL

Introductie - Urban Search and Rescue

In Nederland is in 2002 een speciaal "Urban Search and Rescue" team opgericht: USAR.NL. Dit team kan nationaal of internationaal worden ingezet om mensen in rampgebieden op te sporen, te redden en te assisteren (figuur 1).

Figuur 1: USAR.NL redt iemand tijdens een oefening in Tsjechië (2008).

De uitzending en inzet van USAR.NL is een zeer complex en dynamisch proces. Communicatie en informatie zijn essentieel, maar vaak lastig te coördineren door de wisselende omstandigheden en gelijktijdige inzet van meerdere teams op verschillende plekken in een rampgebied. In deze opdracht worden informatie tools ontwikkeld die USAR.NL bij inzet kunnen ondersteunen. De focus hierbij ligt op de interactie tussen mens en machine.

Groepsopdracht Sebastiaan, Alex en Christian:

Tijdens deze opdracht ontwikkelen jullie een prototype waarmee essentiële informatie verzameld, opgeslagen en gepresenteerd wordt. Hiervoor zijn verschillende technieken beschikbaar, zoals de Diamond Surface table, de Iphone, de Ipad en RFID. Elk van jullie heeft zijn eigen specifieke opdracht, maar uiteindelijk zal er één werkend prototype moeten komen, waardoor er veel overlap zal zijn tussen jullie individuele opdrachten.

Opdracht Sebastiaan: De automatische verzameling van informatie

In deze opdracht wordt gekeken hoe de automatische verzameling van informatie kan bijdragen tot een veilige en efficiënte inzet van USAR.NL. Op dit moment gebruikt USAR.NL alleen gesproken communicatie voor informatie uitwisseling. Nieuwe technieken moeten het mogelijk maken automatisch informatie te verzamelen, zodat er beter en sneller een overzicht van de situatie opgebouwd kan worden.

Tijdens deze opdracht werk je aan het gebruikt van RFID's voor verschillende toepassingen, zoals:

- In- en uitchecken van personeel. Doormiddel van een in- en uitcheck systeem kan belangrijke informatie beschikbaar komen, bijvoorbeeld over werk en rust tijden.
- In- en uitchecken van materieel. Afhankelijk van het werk wat de reddingswerkers gaan doen, nemen ze speciale gereedschappen mee. Omdat de verschillende teams op verschillende locaties werken is het belangrijk te weten met welk team het materieel meegaat. Tevens is de registratie van het gebruik en eventuele defecten goed te gebruiken in het onderhoudsproces.
- Het opslaan en toegankelijk maken van de verzamelde informatie en het helpen ontwikkelen van een applicatie waarin de verzamelde informatie gebruikt kan worden.

Opdracht Christian: Het aanbieden van informatie

De beschikbare informatie moet op een toegankelijke en gebruiksvriendelijke manier aan de verschillende USAR.NL functionarissen aangeboden worden. Tussen deze functionarissen bestaat echter een groot verschil in informatiebenodigdheden (en het benodigde abstractieniveau) en gebruikerscontext. Tijdens deze opdracht werk je aan ontwerpen voor drie type gebruikers:

- De staf in de commando-post. De staf maakt voornamelijk gebruik van een surface table.
- De operationeel commandant. De operationeel commandant is veel onderweg. Hij overlegt met de plaatselijke autoriteiten, is veel in het basiskamp te vinden en stemt continu af met de commando-post. Een Ipad zou hiervoor gebruikt kunnen worden.
- De commandant reddingsteam. Deze persoon heeft de leiding over een relatief zelfstandige reddingsteam. Hij verzorgt de communicatie met de staf, maar heeft hiervoor niet vaak zijn handen vrij gezien zijn 'hands-on' rol in het team. Voor deze gebruikersgroep denken we aan het gebruik van een Iphone.

Opdracht Alex:

De commandanten van de reddingsteams hebben de leiding over relatief zelfstandige reddingsteams en hebben een belangrijke 'hands-on' rol in het team. Juist omdat deze teams zo zelfstandig zijn en de reddingswerkers ter plekke weinig tijd hebben om informatie door te geven, mist de staf vaak essentiële informatie. Tijdens deze opdracht werk je aan het ontwerpen van nieuwe, intuïtieve concepten om belangrijke informatie op te slaan in digitaal formaat.

- Op het moment is het al mogelijk foto's te maken met een Iphone en deze, gekoppeld aan GPS informatie, direct te delen met de staf. Tijdens deze opdracht wordt er gekeken hoe er extra informatie aan deze foto's toegevoegd kan worden, bijvoorbeeld door op de foto te tekenen, symbolen te gebruiken of een tekstboodschap mee te sturen. De nadruk ligt hierbij op een gebruiksvriendelijke interface voor de commandant van het reddingsteam.
- De operationele context waarin de commandanten van de reddingsteams werken stelt hoge eisen aan de interface. Tijdens deze opdracht zal er ook gekeken worden naar een nieuwe manier van interactie om informatie in te voeren. Met behulp van bewegingssensoren kunnen gebaren herkend worden, zodat er ook input mogelijk is wanneer er bijvoorbeeld handschoenen gedragen worden.

Figuur 2: Het gebruik van een iPhone om op een eenvoudige en toegankelijke manier data op te slaan in het veld. De reddingswerker voegt extra informatie toe en zendt dit naar de controle post.

Naam	:	Marc Grootjen
email	:	marc@grootjen.nl
Telefoonnummer	:	0346356314 of 0647484098
Meer informatie op	:	http://mmi.tudelft.nl/colldist/index.php/Colldist

Appendix B

Report Orientation Phase

The following pages contain the report of the Orientation Phase. This report extends the Plan of Approach, which was written earlier.

Chapter 1

Introduction

In this plan of approach, we will first give a description of the company and some background information about the project. In Chapter 2, we will give a description of the assignment Next, we will describe our approach to the problem in Chapter 3. In Chapter 4, we will describe the project setup. Finally, we will present a quality assurance plan in Chapter 5.

1.1 Description of the Company

TNO Defence, Security & Safety is a core department of the TNO research institute. This department focuses on researching new methods for preventing crime, accidents and terrorism. This research is done in cooperation with the Dutch Ministry of Defence, the aerospace sector, the maritime sector, and the defence industry, in order to develop new technological solutions.

1.2 Project background and motivation

This BSc-project is a part of the "Collaboration at a Distance" project, a collaboration between the TU Delft and a TNO business unit. The "Collaboration at a distance" project operates in the domain of Urban Search and Rescue, especially the Dutch "Urban Search and Rescue"-team (USAR-team). The project aims to improve the collaboration between people at different location, something that occurs frequently when different USAR teams are deployed at different locations. The project supports the USAR.NL-team with technological solutions to improve the safety, efficiency and effectiveness of the USAR-missions.

Much research has been done at TNO and the TU Delft, in which domain research has been conducted, and in which several proposals were made for systems that support the USAR.NL-team. For example, a software system called RubbleViewer was created at the TU Delft to visualise the images produced by Unmanned Aerial Vehicles of the disaster area.

This BSc-project aims to create a presentable system for the USAR.NL-team, building forth on the research that has been done in this domain by TNO and at the TU Delft.

Chapter 2

Project Description

In this section the domain of the project will be described, we will be looking at the problem definition, the description of the project, the goals of the project, preconditions and the implications of product failures.

2.1 The Client

Our client is TNO Defense, Security and Safety, which has chosen USAR.NL as its domain of interest and which has good connections with the USAR.NL organization.

2.2 Contacts

Our TNO contact person:
Ir. M. Grootjen
marc@grootjen.nl
+31 (0)15 27 86331

Our TU Delft contact person:
Drs. ing. T.E. de Greef
T.E.deGreef@tudelft.nl
+31 (0)15 27 88543

2.3 Problem Definition

An important concept in a disaster situation is collaboration at a distance, as USAR teams are spread out over the disaster area. The rescue teams are all in different areas of operations, and the base camp is located also in a different place. The command group is usually travelling between the base camp and local, regional, or national organizations. In addition to the teams in the field, there are also teams in the Netherlands working to coordinate the transportation of personnel, equipment, and other supplies to the disaster area. Since the teams are so widely spread, it is very difficult to communicate, gather, and distribute information, which is essential to the success of the rescue missions. Furthermore, as the amount of teams out in the field increases, so does the complexity of collecting and the distribution of that information to the right people. Moreover, the rescue teams work autonomously, where in some cases not all information is available to the other teams. All these factors cause problems for the coordination of information exchange, for which we are going to try to find a solution.

2.4 Project Description

During this project, we will be developing a prototype system that will be able to gather, save, and present important and essential information to the rescue workers in a effective manner, ensuring that

the right people get the right and relevant information. In order to achieve this, we make use of different technologies and different platforms. Our clients were thinking that the iPad, iPhone, Microsoft Surface Table, and RFID chips would be suitable.

This project is divided up into three different parts:

1. **Information input** With the increasing amount of information being gathered by the rescue teams, they must be able to place relevant information in the system, to allow for easier distribution of information. At the moment it is already possible to take a photo and combine this with GPS information. During this project, we will be looking at different ways for rescue workers to add additional information, for example the use of symbols, text comments, or drawings on photos or maps. The use of video or sound is also possible for when a rescue work is unable to use the touch screen. This allows the rescue workers more freedom to pass on precise information about the situation. Furthermore, new ways of interaction and input of information will be considered. For example gesture and movement recognitions. When a symbol is spray painted onto a wall, the movement made is recognized, or if one moves the iPhone in a particular way it will trigger a command.

2. **Automatic gathering of information** This subsystem is concerned with information collecting using RFIDs. First, we analyse which types of information can be collected using RFID tags, and what the capabilities are of the different RFID-systems on the market. Second, we investigate which RFID-systems can be used to collect the required information. Third, the software to collect this information can be designed and implemented. Then, the information can be presented to different users in the applications that are developed in the other subsystems for the iPhone and iPad. For example, the staff team can see which teams are available for deployment on a particular site, as well as whether the required equipment is available in the camp. The support team can see which equipment is defective and needs to be repaired, as this is registered in the system. The commander is presented relevant statics of team deployment and about the equipment, to see if there are any problems that may arise in the (near) future.

3. **Information Presentation** The part of the assignment concerned with information presentation will focus on providing information to three different types of users, namely the operational commander, the commanders of the rescue teams and the staff group. The current iPhone application will be improved by providing methods to represent the new information that commanders of individual rescue teams can input using their own iPhone. Whether these commanders need the infromation provided by the RFID chips will have to be investigated.

   This information obtained by RFID chips can also be used on an iPad, along with information provided by the commanders of individual rescue teams. This application could provide the user with an easy to read overview of the current situation, allowing anyone to see the progress of rescue teams and the status of equipment and personnel.

   Work on an application for the staff group using the Microsoft Surface Table has already begun. If our schedule will allow it, the possibility to integrate this application into our project will also be explored.

To properly implement part three of the assignment, the presenation of information, the other two parts are required. Without these, the application will not have any information to display. On the other hand, the information gathered by parts one and two will not be of any use to a USAR organization without a proper method of presentation.

2.5 Goal

By creating these prototypes (which will be used in multiple platforms), the goal is to improve the information exchange, by making the information not alone easier and faster to interpret but also simpler to collect, save and present. This makes for a much more effective and efficient way for distributing information. A more effective and efficient information exchange allows the rescue workers to be better informed, and thus making them more effective and efficient.

2.6 Products produced

At the end of this project, a prototype product will have been developed that will work on different platforms:

- An iPhone application that will allow rescue workers to easily add extra information to the map on the main screen, or on photos that have been created. Also, an interface will be developed to present relevant information in an easy and understandable manner.
- The iPad application will contain some extra features, in addition to the functionality of the iPhone application. This is due to the fact that the iPad has different users it will be used differently compared to the iPhone. It could be used to improve the administration efficiency and effectiveness. RFID chips could, for example, be used to keep track of the personnel and equipment. Also, information will be presented differently, to take advantage of the different hardware specifications, like the larger screen.
- A Microsoft Surface Table application falls outside the scope of our project. There is, however, another project running simultaneously with this one. The idea is, that a working information exchange system is created between the iPhone/iPad applications and the Microsoft Surface Table. With an interface that has the same (as far as possible) look and feel as the the iPad and iPhone interface.

2.7 Preconditions

For our application to work we define some preconditions about the disaster area:

- There is internet present (to be able to use google maps). This can be either through direct internet connection or via an ad hoc network.
- GPS is functioning

2.8 Implications of product failure

In a disaster situation, the stakes are very high. A decision could mean the difference between life and death. That is why information exchange is so important: The more important/relevant information is available, the better a rescue worker is informed, and able to make the right decision. The implications of our product not functioning as it should, or not working at all, are very high. For example, if the application claimed that there was a victim in a certain area, where there isn't a victim, rescue teams will be wasting their precious time searching for this victim.

Chapter 3

Approach

This section will describe the methods and techniques used to reach our goal, as well as the documents created and the estimated time to create them. Then, this information will be used in a detailed planning of the entire project.

3.1 Methods

During this project, we will use the situated Cognitive Engineering methodology (sCE) [1], an iterative process using knowledge about human factors in development. Three different types of issues, namely operational demands, human factor issues and technological design issues can be addressed by collaboration with experts from multiple disciplines. This will lead to a requirements baseline, which will be evaluated and refined during the project. This evaluation will be conducted using expert reviews from our contacts described in section 2.2. If possible, a review with USAR.NL members will take place towards the end of the project.

3.2 Techniques

In order to create the requirements and design documents, we need to use a couple of techniques:

- Use cases, specifying the interaction between USAR members and the system in detail [2]. A use case will focus on the events after the user has performed an action.
- Scenarios, stories about people and their activities [2]. These will provide an overview of the way USAR members interact, both with the system and between themselves.
- Storyboard, containing a series of images showing the different screens a user will see when operating the system.
- Object Model, a generalized version of the class diagram.
- Dynamic models, such as sequence and state diagrams. The former describes the interaction between the classes of a system (and possibly the user), the latter describes the different states and transitions of the system.

3.3 Proceedings

Before the implementation of the actual applications, some documents describing the design will be required:

Requirements Analysis Document (RAD) This document contains the requirements the system should meet, along with scenarios, use case descriptions, an object model and storyboards. The current implementation of the system will also be analyzed. We estimate it will take approximately 5 to 6 days to create this document.

Architectural Design Document (ADD) This document contains a class diagram with several subsystems. The databases to be used by the application will also be described. We estimate it will take around 3 to 4 days to create this document.

Technical Design Document (TDD) A more specific version of the ADD, containing the classes of the system and the methods and variables of these classes. We estimate it will take about 2 days to create this document.

Test and Implementation Plan (TIP) This document will contain a prioritized list of the requirements, according to the MoSCoW principle. A detailed planning of the implementation phase and a plan to test the software will also be included. We estimate it will take approximately 2 days to create this document.

3.4 Planning

See appendix A for a detailed planning of the project.

Chapter 4

Project setup

This chapter will first describe the stakeholders involved in this project. Second we will provide practical information about this project, such as people involved, working hours and project duration. Last we will provide information about the facilities used.

4.1 Stakeholders

The stakeholders involved in this project are:

- TNO Defence, Security and Safety, the company we're doing this internship at.
- USAR.NL, the organisation we're developing the system for.
- TU Delft, the university for which we're doing this project as a BSc-project.

4.2 Information

Our team will be working on this BSc-project from Monday until Friday from 09:00 - 17:30 in EWI room HB10.250 if available. Otherwise we will work in Building 35 at the Cornelis Drebbelweg. The expected deadline for the final version of our system is 2 July 2010. We will have regular meetings, at least once a week, with Marc Grootjen (TNO) and Tjerk de Greef (TU Delft), who are also the domain experts for the USAR-field.

4.3 Facilities

The room we will work in is EWI HB10.250, which is available most of the time during the course of this project. We will also need at least one, preferably two iPhones and iPads for development purposes, and at least two, preferably three MacBooks to develop the systems for the iPhone and iPad on.

Chapter 5

Quality assurance

In this chapter, we describe what measures will be taken to ensure that we deliver a high-quality product. These measures include minimum requirements for the documentation, testing, and evaluation. We will also describe what version control software is used, and which pilots will be produced.

5.1 Documentation

The documentation consists of two parts: The user manual and the documentation of the software itself.

A separate user manual will be created for each platform our software will run on, to reflect the differences in appearance and functionality. The manual will explain all functionality of the system to the intended user.

The documentation of the software will include comments in the source code itself. These comments include a general description of the functionality for each method and class, pre- and postconditions for each method, class invariants for each class, a specification of the input variables each method takes, and a description of the return value for each method. Other documentation that will be produced, such as state machine specifications, control flow diagrams, or decision tables, will be included in a separate document.

5.2 Testing

We will employ a test-driven approach to implementing the software. For each class or method to be implemented, we will first create functional and structural test cases to test the behaviour of that method or class, and then implement the method itself. In doing this, we aim to achieve a high test coverage of the software we create.

The amount of testing time devoted to a particular class or method depends on the results of the risk evaluation that will be conducted. When creating structural test cases, the critical subsystems will be tested more thoroughly than the other subsystems that have a lower risk. The subsystems that have a moderate risk will have at least statement coverage. The critical subsystems will have to satisfy the branch- and node adequacy criteria. More strict coverage criteria, such as Modified Condition/Decision Coverage, will be used if appropriate.

5.3 Version Control

For version control we use the open-source collaboration software system Subversion. This will automatically save all versions to the central server, allowing for (partial) roll-backs if necessary.

5.4 Evaluation

Formative evaluation will take place at predefined moments during the course of this project. At the end of this project, summative evaluation will take place.

During formative evaluation, we will evaluate our progress with respect to milestones and planning, as well as evaluating the team process, whether or not we have met our goals, and the re-evaluation of risks during this project.

During summative evaluation, we will conduct a user acceptance test, we will evaluate our targets concerning the implemented features and the planning, and we will evaluate the team process. We will also evaluate the problems encountered during this project, in order to prevent those problems from happening in future projects.

Chapter 6

Scenarios

In order to illustrate the interaction between the different parts of our assignment, we will use several scenarios. These scenarios describe a series of events and the actions the people involved will take. For the scenarios, we will assume the search and rescue team has a fully established base camp.

Scenario 1 A search and rescue team is needed at work site 7. The staff group can see the available teams and equipment by using the iPad application. Team Alpha is available along with the equipment they require. They will be dispatched to work site 7. When leaving the base camp, both the personnel and equipment will be registered using RFID chips.

Upon arrival at the work site, the team can asses the situation. There appear to be three victims trapped in a collapsed building. When using his spray can to indicate the position of these victims, the commander's iPhone will automatically store the location of the victims. He can also use his iPhone to submit several images of the area with some remarks.

After the team has rescued these victims, their commander can update their status using his iPhone. The staff group in he base camp can see this information and know they do not need to send another team to replace team Alpha.

When returning to the camp, the search and rescue team and their equipment will once again be registered using RFID chips. Any defective items can be registered manually.

The support team can see whenever a piece of equipment is broken, or likely to be in bad shape due to heavy usage. They will try to repair these items. When this is not possible, the staff group will try to have these items replaced.

When the commander returns from his meeting with the local authorities, he can use the iPad to check up on the operations within the camp and see any serious issues he needs to address.

Scenario 2 < Search and rescue team Alpha has been dispatched to work site 7 >

< Search and rescue team Alpha has assessed the situation >

The team has managed to rescue two out of three victims, but is unable to reach the third victim. Their commander updates the information using his iPhone to represent the new situation. They have been working for a long time and another team will have to take over. The staff in the camp can use their iPhone to see that team Alpha's shift has nearly ended. Team Beta is available and will be dispatched to the work site 7.

Upon arrival the commander of team Beta is able to learn about team Alpha's progress quickly by using their iPhones and looking at the map showing the victims. Search and rescue team Alpha can return to base, while team Beta will continue to search for the last victim.

< Search and rescue team Beta has rescued the last victim and update the information >

< Search and rescue team Beta returns to the base >

< The support group repairs or replaces any broken equipment >

Planning IN3405 Bachelorproject				
	Legend			
	PvA	Plan van Aanpak		
	OV	Oriëntatieverslag		
	RAD	Requirements Analysis Document		
	ADD	Architectural Design Document		
	TDD	Technical Design Document		
	TIP	Test & Implementatieplan		
Weeknr	**Date**	**Action**	**Deadline**	**Deliverable**
1	19-4-2010	Make Planning / Start Weblog		
	20-4-2010	PvA		
	21-4-2010	PvA		
	22-4-2010	PvA / OV	PvA finished	PvA
	23-4-2010	OV	Concept OV finished	
	24-4-2010			
	25-4-2010			
2	26-4-2010	RAD/Prepare for brainstorm-session		
	27-4-2010	RAD/Prepare for brainstorm-session		
	28-4-2010	Brainstorm-session		
	29-4-2010	OV / RAD	OV finished	OV
	30-4-2010	Koninginnedag		
	1-5-2010			
	2-5-2010			
3	3-5-2010	RAD		
	4-5-2010	RAD		
	5-5-2010			
	6-5-2010	RAD		
	7-5-2010	RAD		
	8-5-2010			
	9-5-2010			
4	10-5-2010	RAD / ADD	12:00 RAD finished	RAD
	11-5-2010	ADD		
	12-5-2010	ADD		
	13-5-2010	Hemelvaartsdag		
	14-5-2010	ADD	17:30 ADD finished	ADD
	15-5-2010			
	16-5-2010			
5	17-5-2010	TDD		
	18-5-2010	TDD	17:30 TDD finished	TDD
	19-5-2010	TIP		
	20-5-2010	TIP	17:30 TIP finished	TIP
	21-5-2010			
	22-5-2010			
	23-5-2010			
6	24-5-2010	2e Pinksterdag		
	25-5-2010	Start implementation		
	26-5-2010			
	27-5-2010			
	28-5-2010			
	29-5-2010			
	30-5-2010			

Weeknr	Date	Action	Deadline	Deliverable
7	31-5-2010			
	1-6-2010			
	2-6-2010			
	3-6-2010			
	4-6-2010			
	5-6-2010			
	6-6-2010			
8	7-6-2010			
	8-6-2010		17:30 First runnable finished	First runnable
	9-6-2010			
	10-6-2010			
	11-6-2010			
	12-6-2010			
	13-6-2010			
9	14-6-2010			
	15-6-2010			
	16-6-2010			
	17-6-2010			
	18-6-2010			
	19-6-2010			
	20-6-2010			
10	21-6-2010	Bugfixing / Final Report	09:00 No New Functionality	
	22-6-2010	Bugfixing / Final Report		
	23-6-2010	Bugfixing / Final Report		
	24-6-2010	Bugfixing / Final Report		
	25-6-2010	Bugfixing / Final Report		
	26-6-2010			
	27-6-2010			
11	28-6-2010	Bugfixing / Final Report		
	29-6-2010	Bugfixing / Final Report		
	30-6-2010	Bugfixing / Final Report		
	1-7-2010	Bugfixing / Final Report		
	2-7-2010	Bugfixing / Final Report	17:30 Final Report finished	Final & Report
	3-7-2010			
	4-7-2010			
12	5-7-2010	Presentation		
	6-7-2010	Presentation		
	7-7-2010	Presentation	17:30 Presentation finished	
	8-7-2010	Presentation		
	9-7-2010	Presentation	Final DEADLINE Presentation!!!	

Bibliography

[1] M.A. Neerincx. Situated Cognitive Engineering for Crew Support in Space. *Personal and Ubiquitous Computing*.

[2] M.B. Rosson and J.M. Carroll. *Usability engineering: scenario-based development of human-computer interaction*. Morgan Kaufmann Pub, 2002.

Appendix C

Requirements Analysis Document

1 Introduction

This Requirement Analysis Document follows the Situated Cognitive Engineering approach [**?**] and describes what kind of conditions our system will have to work in and what it will be able to do. This introduction will give an example scenario which illustrates the current way of working and possible improvements that could be made. Previous work on this project will be described briefly as well. Section 2 presents a work domain and support analysis, specifically looking at the Operational Demands, Human Factors and Envisioned Technology. Section 3 gives the (non) functional requirements, claims and use cases. Finally, Section 4 describes the user interface design using storyboards.

1.1 Scenario

An earthquake measuring 7.4 on the Richter scale struck Turkey and USAR.nl has been called in for assistance. The entire organisation flew in with their equipment within the first 24 hours after the earthquake. Directly after landing at the airport, the operational commander has a briefing with the local authorities. Main topics of this briefing are the assignment of two work sites where USAR.nl will be working and the location where the base camp will be build. Furthermore, the commander gets a full update about all relevant circumstantial information, like possible threats of civilians plundering, embassy contacts, transport, supplies, weather information etc. The operational commander uses notes and a map of the area to record this important information.

The commander of the support group is in charge of building the base camp. The entire USAR.nl group unloads the plane and starts building the tents for the different groups, e.g. specific tents for the rescue groups, a medical tent and staff tent that serves as command post. All the equipment is stored at a central location in the camp.

The work sites are located eight and fifteen kilometres from the base camp. The staff decides that team Alpha and Bravo are the first teams that start searching for survivors, so they ask the operational commander to look for the commanders of the rescue teams. Both commanders are briefed in the command post using a map of the area and all information that was provided by the local authorities. After the briefing they will pick the equipment they need, which is administered by the support team, they brief their teams and get on the transportation to the rescue site.

Team Alpha arrives at their work site and starts exploring the area. A few minutes later a partially collapsed office building is found, probably with live victims trapped inside. Another international USAR team already did a quick search of the building but did not find any live victims. They have spay painted a symbol on one of the walls to indicate this. However, locals insist that they have heard somebody, so the commander decides to do a scan with a search dog.

Team Bravo arrived and also starts to explore the second site, they quickly find a number of deceased victims and, after a few minutes, the first signs of life from under a collapsed house. It seems to be a family, consisting of four people. Through communication with one of the family members they find out that at least three of them should be alive.

After the thorough scan of team Alpha, they still did not find any signs of life. They want to proceed searching at the next building, but the local people, who still think their relatives are alive, start arguing. Because the situation is getting more and more hostile, team Alpha has to go back to the base camp.

The staff group in the base camp is tracking the progress of the operations. They manage the mission from the command post and therefore need to keep up to speed with what is happening out in the field. This is a difficult task, especially because they are at a distant location from the work sites. After 7 hours of work, team Bravo managed to rescue two of the family members. The commander of the rescue team gives a situational report to the staff team (through radio or telephone) to update them of the situation. They have been working for a long time and another team will have to take over. The staff decides to send team Charlie. The commander is briefed in the command post, decides to take additional equipment after contact with the commander of team Bravo and leaves for the site. After eight hours team Charlie arrives at the work site. Team Charlie needs a good overview of the situation to take over command. The rescue group commanders get together and discuss the most important events of the last 8 hours and future planned events. Important aspects like the structure of the collapsed building under which the victim is trapped are discussed extensively, and during the first actions of team Charlie, team Bravo stays to see if everything is going well. After half an hour team Bravo returns to the base camp.

Then, an aftershock occurs. This is a stressful situation for the staff group, because members of USAR.NL might be in danger. Team Bravo is still on the way to the base camp and team Charlie is working in the collapsed building. Both teams are contacted and report that everyone is fine. Team Charlie needs to check the stability of the building they are working in. A staff member calls in the structural specialist for assistance.

When arriving at the base camp, the commander of team Bravo debriefs in the command post. The equipment that was brought back needs to be checked for defects by the support team.

1.2 Previous Work

Since this is an on going project there has already been developments and prototypes that have been made. There has been a prototype iPhone application made where the user is able to view a map (in the beginning centralizing at the users current position) and navigate through it. The user is able to centeralize the map on his/her current location. The creation of a victim is possible, adding extra information and making pictures. It is also able to access the create victim screen through the use of gesture recognition, where the user moves the iPhone away from himself very quickly.

There has also been server-client protocol made to allow communication between the iPhone and a server. This is because data from the iPhone is saved and placed on the a server so that other applications are able to access and use the information.

2 Work Domain and Support Analysis

To properly formulate the requirements for the system, a work domain and support analysis needs to be made to get a better understanding of the work environment. This analysis is based on a paper written by De Greef et al. [3] The work domain and support analysis can be split up into three sections: Operational Demands, Human Factor Knowledge and Technology Design Space.

2.1 Operational Demands

An analysis of the operational demands is performed, leading to a better understanding of the organization.

USAR.NL operates in man-made or natural disaster areas. As a result, the local infrastructure can often be severely disrupted and will require much time to be operational again. Therefore, a USAR team should be self-sufficient and highly mobile, as the USAR team has to be able to operate anywhere in the world.

USAR.NL is a hierarchical organization, consisting of several groups, namely the command group, the staff group, the support group, and the search and rescue teams. Figure 1 shows an overview of the command hierarchy.

The command group is in charge of the entire USAR.NL team. They communicate with the staff team and external organisations, such as USAR-teams from other countries, the local operational team (LOT) in the Netherlands, the local emergency management authority (LEMA) in the country the disaster occurred, and a UN-office. The commanders need to keep track of the situation in the base camp and the progress at the work sites in order to effectively communicate with these external organisations. In addition, the commander has to be notified about problems with people or equipment.

The staff team manages the operations within the base camp and decides which autonomous search and rescue team to send to which work site. They are the first to know about problems within the base camp or with the teams, and they will try to solve them.

The support group consists of people with many different skills and training, such as a medical team, the technical worker or the ICT-technician. This group performs the tasks that support the operation of the search and rescue teams.

The four USAR rescue teams work autonomously and usually on different work sites a long distance away from the base camp. The teams are assigned to a work site by the staff team, but the team commander decides which collapsed structure to search and which victims are rescued first. As a result, the teams do not know about the situation the other teams are in, which poses a problem when the team is replaced on a work site. The team commander of the relieving team does not know the situation on

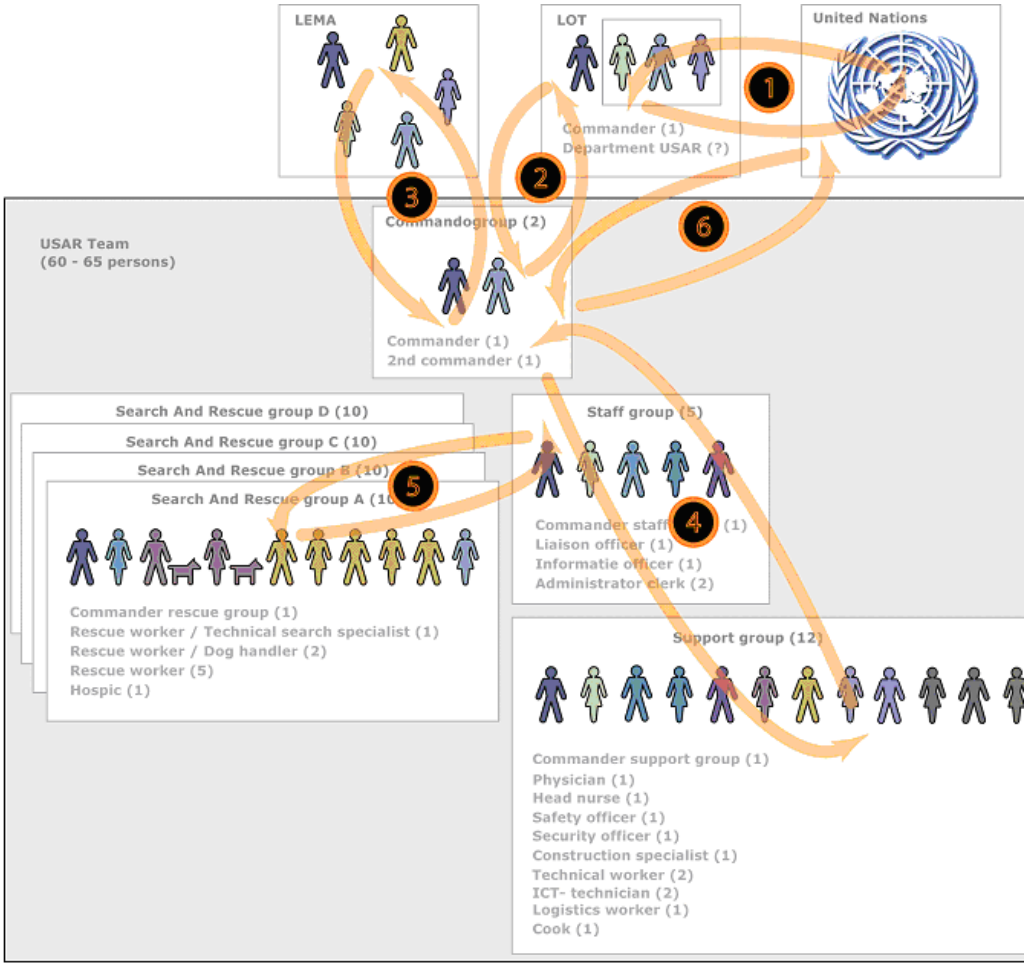

Figure 1: The structure of the USAR.NL team.

the work site. Therefore, an overview of the situation on a work site should be presented to the team commanders on the work site itself, so the relieving of teams will be more efficient.

The commander of a search and rescue team decides which equipment is needed for the current shift at the work site. To make this process easier, there should be a good overview of where the the equipment is located, and whether it is operational and available.

Apart from rescuing victims, a USAR organization aids in reconstructive, medical, and organizational tasks as well.

The USAR team members have different backgrounds in education, employment, and culture. Almost all team members have a job when there is no USAR mission or training. This poses an additional challenge for communication and collaboration between teams.

Also, local culture may differ greatly from mission to mission. Aspects such as economical and social situation, and ethnicity and religion may have an impact on the way the USAR team operates, as communication between the teams and the local people can be obstructed by these cultural differences.

The teams should be able to operate in varying and extreme weather, from tropical to Arctic circumstances. Also, the countries in which the USAR team operates may have unstable governments and plundering of the work sites or the base camp is a concern, since people in the disaster area have lost many possessions. This means all personnel and equipment have to be protected against these possible threats.

Since there are many pieces of equipment that must be shared between teams, and may be lost or stolen, these items must be registered and tracked automatically. Also, the location of the teams must be known to connect that information to the equipment present at the work site where the team is working. The teams can check themselves and items in and out at the base camp and at the work sites, so the

location of those items is known.

The location information must be presented to the different teams, each with different information needs. The members of the commando team will need general information about the state of the teams. The members of the staff team will need to know general information about the different work sites, the teams assigned to those work sites, and the equipment present on those sites. The technical worker will need to know detailed information about the state of the equipment.

1. The location of large equipment and boxes of smaller equipment will have to be gathered with little user input, and will have to be stored automatically.

2. The checking in and checking out of items and people from the base camp and work sites will require the team members to, upon entrance or exit of a work site or the base camp, perform a specific action to enable the system to check in or check out the team members and equipment.

3. The information that is collected will have to be presented to the different teams, with different level of details based on their tasks. The commander of the search and rescue team will need to know information about the work site he/she is currently working on, or will be going to soon. The members of the commando team will need general information about the state of the teams. The members of the staff team will need to know general information about the different work sites, the teams assigned to those work sites, and the equipment present on those sites. The technical worker will need to know detailed information about the state of the equipment.

Objectives

1. The system will keep the commander and staff group informed by collecting and presenting information about personnel and activities, by enabling users to input information about their activities, and by tracking movement of search and rescue teams

2. The system will keep the technical workers informed by collecting and presenting information about equipment, which is gathered with little user input

2.2 Human Factors Knowledge

In addition to the operational demands, human factors should also be taken into account. The most prominent aspects are: worker competencies, collaboration, situational awareness, and time limitations.

Worker competencies look at the competence of a rescue workers, whether they are capable of performing the required tasks. As workers work for long periods of time, they have sleep deprivation and a lot of stress. This can cause the workers' competency to drop, which could lead to mistakes and errors when reading and adding relevant information.

Within the USAR domain, there is much collaboration between different groups within the USAR organization. The most relevant for our system as a whole are:

- Collaboration of the rescue team with the staff team: The staff team coordinate the rescue teams, instructing them where they need to be and giving them background information about the situation at that location.

- Collaboration of the rescue team with another rescue team: When a rescue team is being relieved, the relevant and important information has to be exchanged with the replacing team, so that they are fully informed and able to make correct decisions.

- Collaboration of the rescue team with the support team: When equipment fails and needs repairs, rescue workers need to go to the support team.

- Collaboration of the staff team with the command team: The staff team needs to keep the command team up-to-date about the situation in the field.

In addition, information from various sources within the organization needs to be exchanged.
The human factors specific to the automatic collection of data are specified below:

- The process of checking in and checking out of work sites or the base camp must be convenient for teams to use, since this is an additional task for them to execute compared to the old situation.
- The collected data must be presented in such a way, that situational awareness is improved.
- The information must be presented in a way that the targeted group can understand and process the information within a reasonable amount of time.
- The information must be presented in a way that collaboration between the staff group and the commander of a rescue group is improved, by automating the collection of information about the teams and the equipment the teams take with them.
- The information is collected in an automated way, so human errors and limited human resources in collecting that information will be eliminated

Another human factor that needs to be taken into account in a disaster situation is situational awareness. The situation at a work site is chaotic and constantly changing. When a team relieves another search and rescue team at a work site, they do not know the exact situation at that work site, since all teams operate autonomously. Therefore, the commander of the new team should create situational awareness in a short amount of time. A comprehensive overview of the situation is needed, so that the commander knows all relevant information to base future decisions on. Making the right decisions is vital, because they can make the difference between life and death. Therefore, the information these decisions are based on has to be correct.

Time is an important factor in a rescue mission, since victims are often wounded and do not have access to potable water. A rescue mission only lasts for approximately one week, after which the chances of rescuing more victims have become negligible.

Objectives

1. The system will improve the collaboration at a distance between teams
2. The system will improve situational awareness when switching teams on a work site
3. The system must be easy to use
4. The system will provide feedback to the user when needed to increase the user's trust in the system

2.3 Envisioned Technology

Apart from the operational demands and human factors knowledge, the possible technologies to be used should also be explored. Because different USAR members will have different needs, it will not be sufficient to create a single application on a single device.

The commanders of the search and rescue teams will often be at a worksite. Therefore, they will need a small and portable device.They will also use a spray can to paint several symbols on buildings at the worksite, indicating the location and state of a victim. One platform to be considered would be the iPhone. This device is small and portable, which makes it suitable for someone in need of mobility. In addition, it also possesses an accelerometer to measure the movement of the phone. This could, when attached to a spray can, be used to recognize the symbols at the worksite while they are being drawn.

Apart from the advantages, the iPhone has a couple of disadvantages as well. The small size of the device will decrease the amount of information that can be shown on the screen. The touch-screen itself is capacitive. Whenever someone touches the screen with a finger, the screen will detect a difference in the charge at that position. Because of this, it is impossible to operate the iPhone while wearing gloves. This issue could be solved by weaving conductive thread into the gloves. Finally, the iPhone can only be operated at a temperature between zero and 35 degrees Celsius, a slightly smaller temperature range than most other smartphones.

The operational commander will be on the move very often. He has to meet with the local authorities or USAR teams from other countries, while still keeping track of the situation of his own team. Although he would need a portable device, the screen needs to be large enough to provide a good overview of a situation. The small screen on an iPhone would be insufficient. The iPad however, has a much larger

screen. This greatly increases the amount of information that can be shown at any time. Moreover, the iPad also possesses some additional programming options, such as the ability to use a split screen, create popovers, a small 'balloon' with additional information pointing to an object, and several more interface options. At the moment, all alternatives to the iPad are still in development.

The iPad has a similar screen as the iPhone and therefore, it has to be operated in the same way as an iPhone. In addition, the same temperature restrictions apply. Moreover, the larger size of the device and the lack of a camera will make this device less useful to the commander of a search and rescue team.

An application to provide the staff group with information using a Surface Table is already in development. The advantages of this table are it's large screen size and the ability for multiple people to operate it simultaneously. It is, however, not portable, making is useless to the individual search and rescue teams. An application for this device will not be created by us. We will, however, attempt to integrate the application in development into our project.

We will create an iPhone application for the commanders of the search and rescue teams, because a larger device would be too impractical. An extension to this application on the iPad will be made for the commander of the entire USAR operation, because the larger screen can display more information. The mechanics of the support group could also use the iPad for information collected by RFID chips. Although alternatives to the iPhone could be used, the iPad does not have any alternatives at the moment. Therefore, both the iPhone and the iPad will be used in order to ensure consistency between the two applications.

Gesture Recognition

As mentioned above for the iPhone it might be possible to use gesture recognition to recognize symbols that a rescue worker spray paints on a wall. We have done some research to see how far this is possible.

The iPhone has an accelerometer, which is able to measure the acceleration on the three axes (X,Y,Z). With this information it is possible to calculate the direction the iPhone is moving.

During our research we were able to have an interview with Ir. M. de Hoogh [4] who works at the Technical University of Delft. For the past half year he has been working on a similar program for the wii-mote which is able to recognize numbers. We based our research on his program, because even though it was made for the wii-mote, the input and output, and concepts are the same for the iPhone. The program uses a Hidden Markov Model to be able to recognize gestures.

Method The first step is train the program to recognize certain gestures. The user preforms the gesture multiple times (to create a sample) so that the system can learn how the user makes the gesture. The system will then use the information collected from these sample to make reference points. With these reference point the system will be able to(with probability calculations) guess which gesture that was made.

Problems encountered There were some of the problems that became directly apparent. The program that Ir. M. deHoogh made [4] is very sensitive to the speed and size which the symbol is drawn, also the angle the wii-mote is held influences the gesture recognition. This can differ each time one person makes a gesture, let alone between different people. Thus the program can only be used by the person who trained it.

Problems with using INSARAG symbols is that not all can be drawn in on single motion. Two approaches are possible but both have problems of their own.

It is possible to create an on/off switch for tracking between line segments, but since we only receive raw acceleration data there is no orientation points between these lines segments.

If we decide not to have an on/off switch we are also tracking the movement made between the line segments. This is like drawing a symbol without taking the pen off the paper. Everyone draws symbols differently and with these extra lines (connecting symbols/lines together) can lead to exponential amount of possibilities.

Conclusion of Research Gesture recognition is a very interesting way of placing input into our system, however it is a very big task requiring a lot research and work. Due to its complexities and time constraints it won't be included in the scope of the project.

Having said this, with all the problems mentioned above doesn't mean that this isn't a possible or viable method of input. There are some possible solutions to the afore mentioned problems that could be considered. For example, when the tracking of movement is "off", it is possible to track the movement of the iPhone to get some type of orientation. An iPhone could be given to rescue workers to train and use it for gesture recognition.

There still needs to be more research done and questions answered. For example, is it practical to try to recognize a gesture if you need to put information by hand anyway? A benefit vs cost analysis could also be preformed; how much extra effort will it take for it to recognize the right gesture? Should it recognize the wrong gesture, how much effort is needed to correct this (how often could this occur)? The use of gesture recognition also restricts the rescue workers in how they draw their symbols (what order, speed, angle and size) is this a hindrance to the rescue workers or are they willing to adapt? This are just some of the things that need to be looked into.

RFID

A Radio Frequency Identification (RFID) system usually consists of an RFID-tag, a tag reader with an antenna and transceiver, and a host system [6]. The tag has a small amount of memory, varying from one bit indicating presence to several kilobytes for tags with more capabilities. Most tags contain an identifying, unique code that is transmitted when the tag is activated. This way, the object the RFID-tag is attached to can be uniquely identified.

Types of RFID-technologies

There are many types of RFID-tags, which vary in signal range, transmission frequency, storage capacity, size, and other capabilities [2]. There are, however, three main types of tags: passive tags, semi-active tags and active tags.

Passive RFID-tags do not contain a battery. Therefore, an RFID-reader must emit enough energy through inductive coupling to activate the tag, and enable it to send back its unique identifier. The chip inside a passive tag can, therefore, only operate when being relatively close to an RFID-reader; with general RFID-tags reaching up to 5 metres according to Roberts [6], and state-of-the-art technology reaching up to 10 metres according to Pang [5]. Passive RFID-tags have a virtually unlimited lifetime, as they rely on an external power source to operate. Passive tags are also small (as small as 0.4 by 0.4 millimetres, 0.1 millimetres thick), and cheap ($0.05 - $0.10) [6], making them an attractive option for tracking a large number of items.

Semi-active RFID-tags contain a battery to run the chip's circuitry. The energy needed for communication still comes from inductive coupling with a power source in the reader [6]. The battery power can then be used to power more advanced circuitry, that needs more power than the inductive loop can provide, or circuitry that needs continuous power to operate.

Active RFID-tags contain a battery to run the chip's circuitry and to boost the communication signal with the reader. Active RFID-tags have therefore a much greater range; up to 100 metres [6]. Since the battery provides extra power for the internal circuitry, these tags can be used for monitoring purposes by combining active tags with sensors. By communicating with other tags in an Ad-hoc manner, a large, distributed, monitoring network can be created [6]. With tag-to-tag communication, the range of a tag can be extended to more than a kilometre, by retransmitting signals from other tags towards the RFID-reader. [5] Active tags have a lifetime limited by the capacity of the battery and the power demand of the circuitry. Active tags are also larger than passive tags ($> 1cm^2$), and much more expensive ($1.00 - $100).

Limitations of RFID-technology

RFID-tags are vulnerable to impact damage and sensitive to electromagnetic interference, and therefore cannot be attached to heavy equipment in places where these risks are present. Signal range of most RFID-tags is also impacted severely when placed on metal objects. However, there are active and passive tags that are not impacted by this. Signal range is also impacted when the tags are surrounded by metal objects due to the electromagnetic shielding effect of these objects, so tags cannot be placed within metal boxes.

Objectives

1. An iPhone application will be created to aid the commanders of search and rescue teams
2. An iPad application will be created to aid the command group
3. RFID chips will be used to automatically collect information about personnel and equipment

3 Requirements

3.1 Core Functions

This section describes the core functions of our system. These core functions provide a high-level description of the functionality our system will have.

1. Automatically collect and present information about the location and status of equipment.
2. Provide the users with the right information at the right moment.
3. Provide the users with a clear and easily accessible overview of the situation.
4. Allow manual input of information about victims or dangers at a work site, including detailed information.

3.2 Functional Requirements and Claims

In this section we present the functional requirements for our system and formulate claims about what our system should be able to do when it is finished. These requirements are based on the core functionality of our system.

3.2.1 Automatically collect and present information

The following requirements are made about core functionality item 1.

1.

Functional requirement	The technical worker can view a list of broken equipment.
Claims	The list will be scrollable when it is too large to fit the screen. The technical worker can always see which pieces of equipment are in need of repairs.
Use Cases	UC8.1.

2.

Functional requirement	All large equipment must be uniquely identifiable and registered in the system. The registration number is used as a means to identify a piece of equipment and is the same throughout the system.
Claims	All equipment that needs to be registered has one RFID-tag with a unique code on the tag, which is also included in print on the tag itself. An RFID-tag is associated with only one object or one case of objects.
Use Cases	UC7., UC7.1., UC7.2., UC7.3.

3.

Functional requirement	Statistics about the use of equipment will be generated automatically using information collected by RFID-tags.
Claims	Statistics about the number of times the equipment has been used during this mission are presented. Statistics will be at most 15 minutes out of date, so that when data changes,the technical worker will be able to see that change within 15 minutes.
Use Cases	UC1.2., UC1.1.

4.

Functional requirement	Statistics about equipment usage since the last maintenance, or, if no maintenance has been done yet, since its first use during this mission, can be viewed for each piece of equipment by the technical worker.
Claims	Statistics about the number of times the equipment has been used during this mission are presented. Statistics will be at most 15 minutes out of date, so that when data changes,the technical worker will be able to see that change within 15 minutes.
Use Cases	UC8.2.

5.

Functional requirement	Information about the presence of equipment in the base camp or on a work site is collected automatically using RFID-tags.
Claims	When equipment is checked in or out of the base camp, the information in the database will be updated within 2 minutes after the last object has been scanned, or within 5 seconds after the user has confirmed the list of equipment as "done"
Use Cases	UC8.1.

3.2.2 Right Information at the Right Moment

The following requirements are made about core functionality item 2.

1.

Functional requirement	The Commanders (C1 and C2) are notified of important information (victims, team progress, etc).
Claims	The current location and status of each team can be viewed directly. The commander has access to the same information about victims as the rescue workers (see requirements 2 and 3) .
Use Cases	UC6.1., UC6.2.

2.

Functional requirement	Rescue teams are able to see detailed information about the danger item, including photos.
Claims	Detailed information includes photos and notes about the danger item, if any are available. Detailed information about the danger item is easily accessible through an icon on the map.
Use Cases	UC4.1., UC4.2., UC4.3.

3.

Functional requirement	Rescue teams are able to see the detailed information about victims, including photos.
Claims	Detailed information includes an INSARAG symbol indicating the state of the victim or victims, photos (if any), and notes (if any). Detailed information about the victim is easily accessible through an icon on the map.
Use Cases	UC4.1., UC4.2., UC4.3.

4.

Functional requirement	Rescue teams are able to see the detailed information about the photo (annotations).
Claims	Detailed information (annotations) can be seen in the form of text, symbols and/or drawings.
Use Cases	UC4.1., UC4.2., UC4.3.

5.

Functional requirement	A list of equipment can be viewed by the technical worker, which can be sorted by usage since the last maintenance operation or by number of maintenance operations conducted during this mission.
Claims	The list of equipment contains all registered equipment and is categorised. The list can be sorted by usage since the last maintenance operation or by number of maintenance operations conducted during this mission. The list can be filtered, so the list only displays defect equipment
Use Cases	UC8.1.

6.

Functional requirement	Details of the defect can be viewed by the technical worker.
Claims	Details of the defect include notes, including the dates the notes were made. Details of the defect can be accessed by browsing a list of maintenance reports.
Use Cases	UC8.2.

3.2.3 Provide a Clear Overview of the Situation

The following requirements are made about core functionality item 3.

1.

Functional requirement	Rescue teams can see a detailed map of the work site.
Claims	The current location of the person viewing the detailed map is shown on the map. The map can be browsed by using “swiping” and “pinching” gestures.
Use Cases	UC2.1., UC2.2., UC2.3.

2.

Functional requirement	All victims can be viewed on the map (detailed information can not).
Claims	The victims are shown as icons with INSARAG symbols on the map.
Use Cases	UC2.1., UC2.3.

3.

Functional requirement	All danger areas can be viewed on the map (detailed information can not).
Claims	Danger areas are displayed on the map using icons.
Use Cases	UC2.1., UC2.3.

3.2.4 Manual input of information

The following requirements are made about core functionality item 4.

1.

Functional requirement	A rescue worker can add comments to photos in the form of text, symbols and drawings.
Claims	Text can be attached to a photo as a note, which can be typed using the iPhone on-screen keyboard. INSARAG symbols of victims and symbols of dangers can be placed on the photo itself. Drawings include arrows and circles, which can be placed on the map by selecting the arrow or circle drawing tool, and then use at most two gestures to create the circle or arrow.
Use Cases	UC3.3.

2.

Functional requirement	A rescue worker is able to add a victim with GPS location, comments and photos.
Claims	The INSARAG symbol of a new victim can be selected through a button on the map screen, after which it is placed on the map with as a location the current GPS location. The INSARAG symbol shown on the map is based on the details of the victim, as specified in the detail screen. Comments and photos can be added in the detail screen. An estimate for the time to rescue the victims can be added to the victim.
Use Cases	UC3.1.

3.

Functional requirement	A rescue worker is able to add photos about a dangerous area and/or victim.
Claims	The detail screen of a danger item or victim provides an option to add photos to that victim or danger.
Use Cases	UC3.2.

4.

Functional requirement	A rescue worker is able to add danger areas with GPS location, comments and photos.
Claims	The symbol of a danger area can be selected through a button on the map screen, after which it is placed on the map with as a location the current GPS location. Comments and photos can be added in the detail screen.
Use Cases	UC3.1.

5.

Functional requirement	A rescue worker is able to edit existing comments, victims, photos, etc.
Claims	Comments and details of the victim or danger item can be edited on the corresponding detail screen. Photos can be edited by clicking on the photo in the detail screen, after which the photo is opened and notes and other annotations can be edited. INSARAG victim symbols are updated according to the information on the detail screen.
Use Cases	UC5.1.

6.

Functional requirement	Equipment that has been checked out can be transferred from one team to another.
Claims	Equipment can be transferred when the new team checks out. All equipment will be transferred by pressing four buttons in total.
Use Cases	UC1.3.

7.

Functional requirement	Details about the repair progress of an item, such as the state the repair process is in and notes about the repair process itself, can be added, viewed, and updated by the technical worker.
Claims	Details about the repair progress include information about the defect, recorded in notes, and the date of creation of the note and name of it's creator. Additional notes can be added to an item.
Use Cases	UC8.1., UC8.2., UC10.1. UC10.2., UC10.3., UC10.4.

8.

Functional requirement	New defects can be registered by the technical worker.
Claims	All equipment has an 'in working state' or 'defect' status. The technical worker can change the status of a piece of equipment from the detailed view of this item.
Use Cases	UC9.2.

9.

Functional requirement	Equipment can be added and removed from the catalogue by a technical worker.
Claims	The technical worker can assign the newly added piece of equipment to a category.
Use Cases	UC9.1., UC9.3.

3.3 Non-functional requirements

In this section, we will describe the non-functional requirements. These will contain the criteria that the system will have to meet when delivered and will lead to several new claims.

3.3.1 Reliability

It is important that the system is reliable, as the loss of data or inaccessibility of the data will result in much work needing to be re-done. In a time-critical environment like a disaster zone, where the USAR.NL-teams will operate, every hour lost due to an unreliable system will decrease the chance of survival of the victims trapped inside collapsed buildings and under rubble. Therefore, the system needs to be as reliable as possible. This will lead to the following claims:

1. A failure may not occur more than once per mission.
2. An error may not occur in more than 5% of actions. An action is defined as the sequence in a use case.

3.3.2 Time-efficiency

In the current system, a considerable amount of time is spent briefing the next team that will work at a particular site. Also, it is hard for a commander of a search and rescue team to get a coherent overview of the work site. This system will improve the situational awareness of the commander of the search and rescue team, and decrease the amount of time spent on changing teams. This way, the teams will be more effective at finding victims, which will result in more victims being found alive and rescued.

In order for this system to get accepted by the users, the system will, among many criteria, have to save the users time. The time invested in using this system must, therefore, not outweigh the amount of time spent in the current system. This results in one additional claims:

1. The exchange of information between search and rescue teams should be at least as fast when using the system as it is without the system 75% of the time. This claim assumes the application will not lead to the exchange of more information.

3.3.3 User Interface and Human Factors

A clear, intuitive user interface will have to be created for each user and usage pattern. This way, the user will only receive the information and be offered the functionality relevant to the specific task the user is executing at that moment. This will improve usability and learnability.

The user interface will also have to accommodate for use in the field, accounting for the circumstances the users are in. Furthermore, the user interface must be consistent among the different platforms to improve learnability. To support this, two claims can be formulated:

1. A rescue worker can perform at least 50% of all tasks without any explanation.
2. Experience with one of the produced applications will decrease the learning curve of another produced application.

3.3.4 Hardware Considerations

The hardware considerations for the system consist of several claims:

1. Three applications will be developed, one for the iPhone, one for the iPad, and one to scan the RFID-tags of equipment..

2. All three applications will use the same database.
3. RFID-tags and -scanners will be used to automatically collect information.

3.3.5 Performance characteristics

The following performance characteristics have been set as minimum requirements for interaction with the system:

1. Any command that is executed on any platform during normal operation must take at most 10 seconds to complete. Some processes however, have more strict performance demands.
2. Within 5 seconds after choosing to add a new victim or danger, the user is able to start entering information.
3. After selecting a victim, the user can see it's detailed information within 5 seconds/
4. Within 10 seconds after opening the map, the user can see the location of each victim within the current field of view.

3.3.6 Error handling and extreme conditions

1. The user is notified of any errors that may lead to limited or faulty operation, so her or she can take appropriate action to prevent data loss.

3.3.7 System interfacing

1. De iPhone-, iPad-, and RFID-applications connect to a central server.
2. The iPhone and iPad connect to the server using a wireless connection.
3. The RFID-application can either be connected wireless, or using an ethernet connection.

3.3.8 Quality issues

The demands for quality assurance are formulated in Chapter 5 in the Plan of Approach. This chapter contains the following claims:

1. The user manual explains all available functionality.
2. All source code will contain comments explaining it's functionality.

3.3.9 System modifications

No major modifications are made to the current system of spray-painting victim symbols on walls. The process of changing teams on a work site will be supported by the information and overview of the site provided by this system. This system will support the methodology used in the current system, not replace it. The users can always fall back on the current system.

3.3.10 Physical environment

The iPhone application will be used on a work site, where operating conditions vary largely. The system will have to be operable in physically demanding situations, leading to two additional claims:

1. The iPhone will work within collapsed buildings or similar locations.
2. The iPhone can be operated while wearing the standard equipment of a commander of a search and rescue group.

The iPad application will not be used at work sites. Therefore, no extra demands are imposed by the physical environment.

The device the RFID-application will run on will be connected to scanners located in the base camp.

The database system will have to be installed within the base camp, along with the equipment required to set up wireless communication. No extra demands are imposed by the physical environment.

3.3.11 Security issues

Wireless communication must be encrypted using at least WPA-encryption with 128-bit AES encryption key or equivalent, in order to protect abuse. The server will only be accessible from the wired or wireless network connections on site. Support for VPN for access via the Internet could be added in the future.

3.4 Constraints (Pseudo-requirements)

The system should be able to store information, such as user input (e.g. victim data) and RFID data, on a central server in the base camp. This information should then be made accessible and usable by applications running on different platforms, for example the Surface Table and the iPad.

The system should also be able to get its information, such as victims found by other teams, from the central server in the base camp.

Several additional contraints apply to the use of RFID-tags:

- Large equipment and boxes of smaller equipment will have to be equipped with RFID-tags. These will have to be attached to the carrying case of the equipment itself, or to the box the smaller equipment is carried in.
- RFID-tags attached to metal boxes will need to be placed on the outside of these boxes, and must be resistant to the effects the metal will have on the tags.
- The code of the RFID-tags will have to be matched to the item or a list specifying the contents of a box of items.
- The checking in and checking out of items and people from the base camp and work sites will require the team members to, upon entrance or exit of a work site or the base camp, walk within range of the RFID-tag reader on that site for at least the time necessary for the reader to read all tags on the equipment and that member of the team.

3.5 Use case model

The following use case diagram describes the relations between the use cases and the actors, and the relations between the use cases. These use cases have been grouped by the application they apply to.

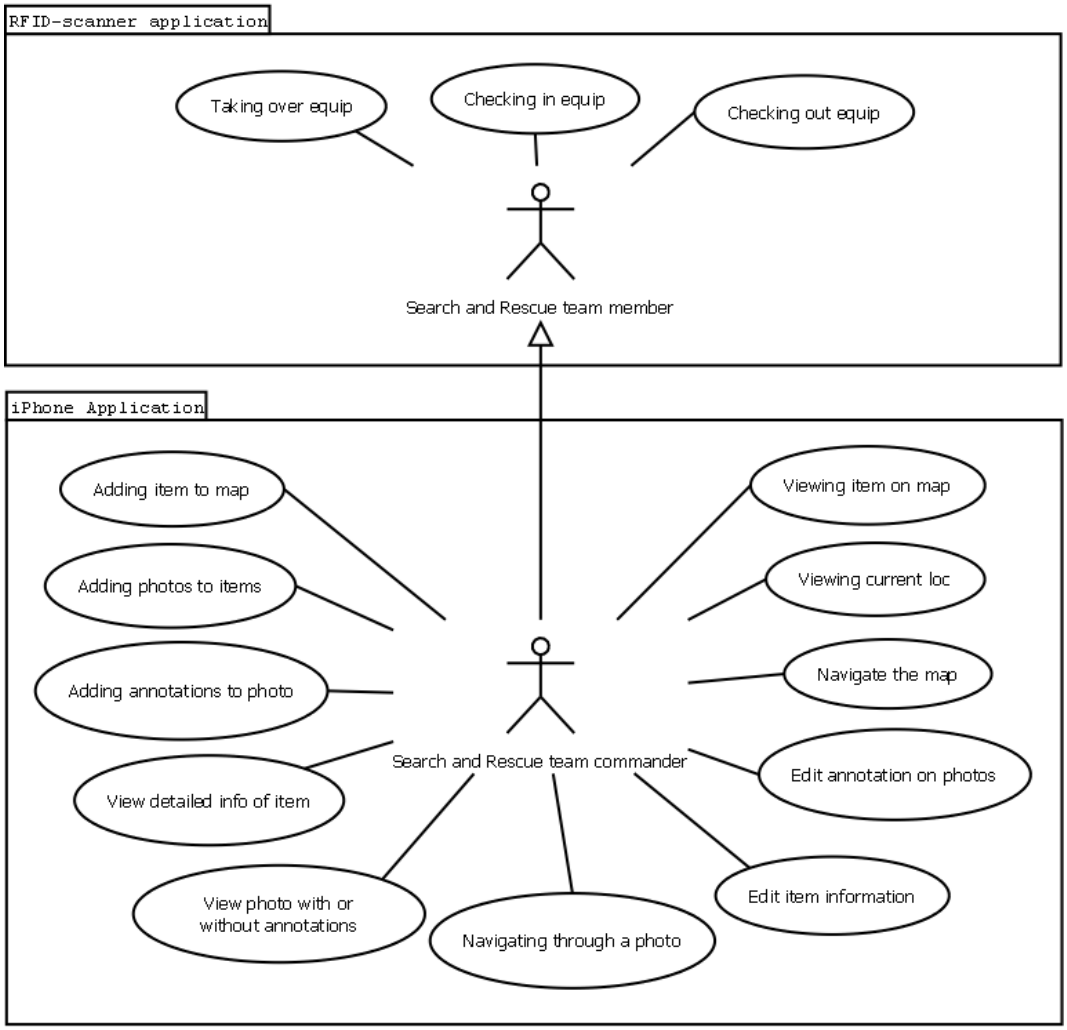

Figure 2: Use case diagram for the iPhone and RFID-application.

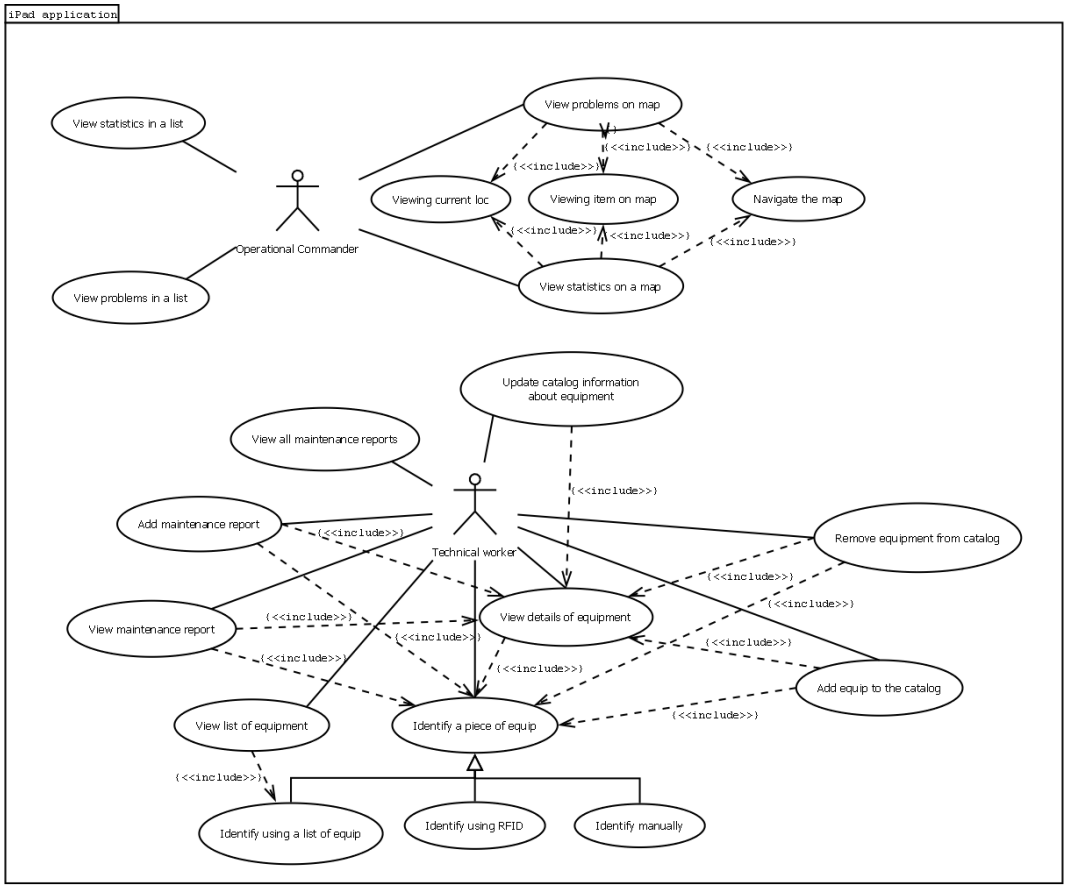

Figure 3: Use case diagram for the iPad application.

1. Checking equipment in and out

1.1. Checking out equipment

Actors: Search and Rescue team members.
Goal: Registers equipment as belonging to the team that has taken the equipment.
Preconditions: The equipment is not currently registered as belonging to another team.
Summary: The actor wants to take certain pieces of equipment with them. They scan this equipment one by one at a designated scanning point in the base camp, after which the equipment has been registered as taken by the team the actors belong to.
Related use cases: Checking in equipment (1.2.)
Taking over equipment from another team (1.3.)
Steps:

Actions taken by actor	System response
Walks to a RFID console in the base camp and specifies the team the actor is in and the intention to check out equipment on the RFID console.	Signals that the team has been selected and that the system is ready to check out equipment.
Brings the RFID-tag of an unscanned piece of equipment within range of the RFID scanner	Signals that the object has been scanned and checked out.
Repeat above procedure until all equipment that the actor want to take to the work site has been checked out	-

Postcondition: All pieces of equipment the actor wants to bring is registered as checked out of the base camp, now belonging to that team.

1.2. **Checking in equipment**
Actors: Search and Rescue team members.
Goal: To register the equipment that has been brought back by a team.
Preconditions: The equipment is currently registered as belonging to a team.
Summary: The actor has brought back equipment from a work site. They scan this equipment one by one at a designated scanning point in the base camp, after which the equipment has been registered as being back in the base camp.
Related use cases: Checking out equipment (1.1.)
Taking over equipment from another team (1.3.)
Steps:

Actions taken by actor	System response
Walks to a RFID console in the base camp and specifies the intention to check in equipment on the RFID console.	Signals that the system is ready to check in equipment.
Brings the RFID-tag of an unscanned piece of equipment within range of the RFID scanner	Signals that the object has been scanned and checked in.
Repeat above procedure until all equipment that has been brought back from the work site has been checked in	-

Postcondition: All pieces of equipment the actor brought back from the work site have been registered as checked in at the base camp.

1.3. **Taking over equipment from another team**
Actors: Search and Rescue team members.
Goal: To register equipment belonging to a team being relieved to another team that is taking over.
Preconditions: The equipment is currently registered as belonging to the team being relieved.
Summary: The actor wants to take over the equipment present at the location of the relieved team. At a designated point in the base camp, the actor specifies which team they are taking over and which team they belong to.
Related use cases: Checking in equipment (1.2.)
Checking out equipment (1.1.)
Steps:

Actions taken by actor	System response
Walks to a RFID console in the base camp and specifies a team will be taking over from another team.	Shows the options to specify which team is taking over the other team, and a confirmation option.
Specifies the team which is to be taken over from, the team the actor is in.	Signals which team is taking over equipment from which team.
Confirms that equipment will be taken over from the other team.	Confirms that all equipment from the other team has been taken over by the actor's team.

Postcondition: Equipment from the other team has been taken over by the actor's team.

2. View map

2.1. **Viewing a victim or danger item on the map**
Actors: Commander of the search and rescue team or operational commander.
Goals: To allow the actors to view a victim or danger item on the map.
Preconditions: Is on the Map Screen (initial screen of the iPhone application).
Summary: The actors can see the dangers and victims on the map that have been added by themselves or others.
Related Use Cases: Navigate the map (2.3.).
Steps:

Action of the actors	System response to actions
-	Victim and danger items are shown on the map
Navigates the map to see other areas	New victim and danger symbols appear

Postcondition: Victim and danger items are shown on the map.

2.2. **Viewing current location on map**
Actors: Commander of the search and rescue team and rescue workers or operational commander.
Goals: To center the map on the user's current (or last known) GPS position.
Preconditions: Is on the Map Screen (initial screen of the iPhone application).
Summary: The actor can not see his own position on the map any more. By pressing the current position button he centres the map on his last known GPS position.
Related Use Cases: None
Steps:

Action of the actors	System response to actions
Presses the current position button	The map is centred on the user's current (or last known) GPS position

Postcondition: The map is centred on the user's current (or last known) position, indicated by a symbol.

2.3. **Navigate the map**
Actors: Commander of the search and rescue team and rescue workers or operational commander.
Goals: Zoom in or zoom out on the map. Move to another part of the map.
Preconditions: Is on the Map Screen (initial screen of the iPhone application).
Summary: The actor can navigate the map to look at different areas.
Related Use Cases: View the map (2.1.).
Steps:

Action of the actors	System response to actions
Moves two finger closer together on the screen	Zoom out on the map
Moves one finger across the screen	Pan the map into the opposite direction
Moves two fingers away from each other on the screen	Zoom in on the map at the position of the movement

Postcondition: On the map screen.

3. Adding items to the map

3.1. **Adding a victim or danger items to the map**
Actors: Commander of the search and rescue team.
Goals: To allow the actors to place a victim or danger item on the map.
Preconditions: The GPS is functioning; Is on the Map Screen.
Summary: The actor has discovered a victim or danger area and wants to place it in the system. In the Map Screen he will press the add victim or danger button. A Detailed Information screen will appear where he will add more information about the victim. Once finished he will press the Save button.
Related Use Cases: None
Steps:

Actions taken by actors	System response
Presses the add victim or danger button	Detailed information screen appears
Presses a section on the detailed information screen to add information	Allows the actor to add information
Presses the Save button	Saves the added information and places a symbol on the map

Postcondition: The victim or danger item has been added to the map (with detail information).

3.2. **Adding photos to items**
Actors: Commander of the search and rescue team.

Goals: To allow the actors to add photo linked to a victim or danger item.
Preconditions: The Camera of the iPhone is functioning; Is in the Detailed Information Screen of an existing item.
Summary: Once a victim or danger has been made or is being made. from the Detailed Information Screen the actor has the option to take a picture. He will press take photo button and then take a photo.
Related Use Cases: Adding a victim or danger items to the map (3.1.).
Steps:

Actions taken by actor	System response
Presses the take photo button	The camera application of the iPhone starts up
Picture is taken	The picture is saved and linked to the specific item of which the detailed information pertains

Postcondition: Photo is saved and can be seen in the Detailed Information Screen of the item of which it pertains to.

3.3. **Adding annotations to a photo**
Actors: Commander of the search and rescue team.
Goals: To allow the actors to make annotations (drawings) to a photo.
Preconditions: A photo has been taken; Is in the Photo View Screen.
Summary: With the photo that has been taken the actor is able to make annotations (drawings).
Related Use Cases: Adding photo to item (3.2.),
Includes: Navigating through a photo (4.3.).
Steps:

Actions taken by actors	System response
Presses the drawing options	The drawing option is activated
Draws on the photo using his/her finger	Drawings are made on the photo
Presses the Save button	Drawings are saved and linked to the specific item of which it pertains to

Postcondition: Photo is saved and can be seen in the Detailed Information Screen of the item of which it pertains to.

4. View detail information of an item on map

4.1. **View detailed information of a victim or a danger item**
Actors: Commander of the search and rescue team and operational commander.
Goals: To allow the actors to see detailed information about a victim or danger item.
Preconditions: For the information to be viewed, there exists an item (victim or danger); Is in the Map Screen.
Summary: An actor wants to view more detailed information about a victim or danger. From the Map Screen he will select the victim or danger and the Detailed Information Screen will appear. **Related Use Cases:**Adding a victim or danger items to the map (3.1.).
Steps:

Actions taken by actor	System response
Presses the corresponding symbol to get detailed information about a victim or danger	A small pop up will appear with basic information
Presses the more information button in the pop up	The Detailed Information Screen for that item will be displayed

Postcondition: Is in the Detailed Information Screen pertaining to a victim or danger item and is able to view the detailed information.

4.2. **View photo with and without annotations**
Actors: Commander of the search and rescue team and operational commander.
Goals: To allow the actors to see/hide annotations made on a photo.
Preconditions: For the photo to be viewed there is a link between an existing item (victim or danger); Is in the Detailed Information Screen of an item.
Summary: The actor wants to view the photo sometimes with the annotation but also sometimes without. The actor will select the photo he/she wants to see and is able to turn the annotations on and off through a filter button.
Related Use Cases: Adding photo to item (3.2.), Adding Annotations (3.3.)
Extension View detail information of an item on map (4.1.),
Includes: Navigating through a photo (4.3.). **Steps:**

Actions taken by actor	System response
Presses the thumbnail of the photo	Photo will be displayed
Presses the annotation filter button to turn annotations off or on	The photo will be displayed without or with annotations

Postcondition: The actor is able to view a photo with and without annotations with the use of the filter annotation button.

4.3. **Navigating through a photo**
Actors: Commander of the search and rescue team.
Goals: To allow the actors to navigate through a photo, zooming in and out, moving to the right,left, up and down.
Preconditions: For the photo to be navigated, an item (victim or danger) has been made with a linked photo.
Summary: The actor is able to navigate through a photo, zooming in and out, and moving the photo right, left, up and down.
Related Use Cases: View photo with and without annotations (4.2.).
Steps:

Actions taken by actor	System response
Places two fingers down on the screen and move them towards each other	The photo will zoom in
Places two fingers down on the screen and move them towards each other	The photo will zoom out
Places one finger down on the screen and swipes to the right	The photo will move to the right
Places one finger down on the screen and swipes to the left	The photo will move to the left
Places one finger down on the screen and swipes to the up	The photo will move to the up
Places one finger down on the screen and swipes to the down	The photo will move to the down

Postcondition: The actor is able to navigate through a photo by zooming in and out, moving it right,left, up and down.

5. Edit map item

5.1. **Editing a victim or danger item information**
Actors: Commander of the search and rescue team.
Goals: To allow the actors to edit the detailed information pertaining to a victim or a danger item.
Preconditions: For the information that is to be changed, there exists an item (victim or danger); Is in the Map Screen.
Summary: The actor will want to edit information about a victim or danger. He will be able to do this by selecting the item he want to change. In the Detailed Information Screen he will edit the information.

Related Use Cases: View detailed information of a victim or a danger item (4.1.).
Steps:

Actions taken by actor	System response
Presses the symbol of an item (victim or an danger)	The Detailed Information Screen is displayed
Presses the section in Detailed Information Screen to edit	Allows the section to be edited
Presses the Save button when finished	The edited information will be saved

Postcondition: The the edited information is saved and is visible when the Detailed Information Screen is displayed for that particular item.

5.2. **Undo annotations to a photo**
Actors: Commander of the search and rescue team.
Goals: To allow the actors to undo annotations (drawings) to a photo.
Preconditions: A photo has annotations; Is in the Photo View Screen.
Summary: With the photo that has annotations, the actor is able to undo annotations (drawings) made.
Related Use Cases: Adding photo to item (3.2.), Adding annotation to photo (3.3.), Clear annotation to photo (5.3.),
Includes: Navigating through a photo (4.3.).
Steps:

Actions taken by actors	System response
Presses the undo option	Removes the previous drawn annotation
Repeat above procedure to remove more annotation	-
Presses the Save button	Changes are saved and linked to the specific item of which it pertains to

Postcondition: Changes are saved and can be seen in the Detailed Information Screen of the item of which it pertains to.

5.3. **Clear annotations to a photo**
Actors: Commander of the search and rescue team.
Goals: To allow the actors to clear all annotations (drawings) on a photo.
Preconditions: A photo has annotations; Is in the Photo View Screen.
Summary: With the photo that has annotations, the actor is able to clear all annotations (drawings) made.
Related Use Cases: Adding photo to item (3.2.), Adding annotation to photo (3.3.), Undo annotation to photo (5.2.)
Includes: Navigating through a photo (4.3.).
Steps:

Actions taken by actors	System response
Presses the undo option	Removes the previous drawn annotation
Repeat above procedure to remove more annotation	-
Presses the Save button	Changes are saved and linked to the specific item of which it pertains to

Postcondition: Photo is saved and can be seen in the Detailed Information Screen of the item of which it pertains to.

6. View statistics

6.1. **View statistics on a map**
Actors: Operational commander.
Goals: To view the details of a worksite on the map screen.
Preconditions: On the menu screen (iPad only).
Summary: The actor finds the location of a rescue team on a map and can see their statistics and progress.
Related Use Cases: None
Steps:

Action of the actors	System response to actions
Presses the view map button	Change to the map screen
Presses the symbol of a team on the map	A pop over with statistics of the team and progress of the work site appears
Presses anywhere on the map except the pop over	Close the pop over

Postcondition: The maps screen is shown.

6.2. **View statistics in a list**
Actors: Operational commander.
Goals: To view all teams and their collected statistics.
Preconditions: On the menu screen (iPad only).
Summary: The operational commander can see a list of teams and the statistics that have been collected, such as the number of victims saved, and the time they have been working or in the base camp.
Related Use Cases: None
Steps:

Action of the actors	System response to actions
Presses the view personnel statistics button	Change to the view personnel statistics screen
Moves finger upwards on the screen	Scroll down on the list
Moves finger downwards on the screen	Scroll up on the list
Presses the view team button	Change to the map screen and centre the map on the team

Postcondition: The team statistic screen is shown.

7. Identify a piece of equipment

Actors: Technical workers.
Goal: Specifying identity information of a piece of equipment to the system.
Preconditions: None.
Summary: The actor specifies the identity of the piece of equipment to the system.
Generalisation: Identify using a list of equipment (7.1.), Identify using RFID (7.2.), Identify manually (7.3.)

Steps:

Actions taken by actor	System response
Specifies the piece of equipment he/she is looking for in the system	Signals that the object has been recognised by the system.

Postcondition: The piece of equipment has been identified by the system.

7.1. **Identify using a list of equipment**
Actors: Technical workers.
Goal: Specifying identity information of a piece of equipment to the system.
Preconditions: The equipment has been registered in the system.
Summary: The actor opens the list of equipment, searches for the piece of equipment he is looking for in the list, then selects the piece of equipment.
Related use cases: Precondition: View list of equipment (8.1.)
Generalization: Identify a piece of equipment (7.)

Steps:

Actions taken by actor	System response
The steps specified in use case "View list of equipment" (8.1.) are executed.	Shows a categorised view of all equipment.
Searches for the piece of equipment he/she is looking for and selects that item	Signals that the object has been recognised by the system.

Postcondition: The piece of equipment has been identified by the system.

7.2. **Identify using RFID**
Actors: Technical workers.
Goal: Specifying identity information of a piece of equipment to the system.
Preconditions: The piece of equipment has an RFID-tag that can be read by the scanner.
Summary: The actor specifies that he/she wants to scan an RFID-tag, then places the RFID-tag of the piece of equipment within range of the RFID-scanner.
Related use cases: Generalization: Identify a piece of equipment (7.)

Actions taken by actor	System response
Specifies that he/she wants to specify the identity of a piece of equipment using RFID.	Signals the system is ready to scan an RFID-tag.
Places the RFID-tag of the piece of equipment within range of the RFID-scanner	Signals that the object has been recognised by the system.

Postcondition: The piece of equipment has been identified by the system.

7.3. **Identify manually**
Actors: Technical workers.
Goal: Specifying identity information of a piece of equipment to the system.
Preconditions: None.
Summary: The actor specifies that he/she wants to identify a piece of equipment to the system manually.
Related use cases: Generalization: Identify a piece of equipment (7.)

Actions taken by actor	System response
Specifies that he/she wants to specify the identity of a piece of equipment manually.	Signals the system is ready for user input of an identification code in a text field.
Specifies the identification code in the text field	Signals that the object has been recognised by the system.

Postcondition: The piece of equipment has been identified by the system.

8. View information about equipment

8.1. **View list of equipment**
Actors: Technical workers.
Goal: To view a list of equipment.
Preconditions: The equipment has been registered in the system. The home screen of the equipment maintenance application is open.
Summary: The actor views a categorised list of equipment.
Related use cases:

Steps:

Actions taken by actor	System response
(The home screen of the equipment maintenance application is already open, no actions are required)	-

Postcondition: A view with a list of equipment is shown.

8.2. **View details of equipment**
Actors: Technical workers.
Goal: To view the status of a piece of equipment, which need not be present in the place the technical worker is in.
Preconditions: The equipment has been registered in the system.
Summary: The actor specifies the piece of equipment he wants to see the details of, and then views a view with details of that piece of equipment.
Related use cases:
Identify a piece of equipment (7.)
Steps:

Actions taken by actor	System response
Specifies the piece of equipment he/she is looking for in the system, following the steps in use case Identify a piece of equipment (7.), and specifies the intention to open a view with the details of that equipment	Opens a view with the details of that piece of equipment.

Postcondition: A view with details of the specified piece of equipment, including an option to view the maintenance reports about that item, has been opened.

9. Update equipment catalog

9.1. **Add equipment to the catalog**
Actors: Technical workers.
Goal: To register new equipment in the catalog.
Preconditions: The piece of equipment is not currently registered in the system. The home screen is open.
Summary: The actor specifies that he/she wants to register new equipment in the system. A screen is shown, where the actor fills out the required information about the piece of equipment. Then, the actor selects a category of equipment or creates a new category. The information is then saved in the system.
Related use cases: View details of equipment (8.2.)
Identify a piece of equipment (7.)
Steps:

Actions taken by actor	System response
Specifies in the home screen that he/she wants to register a new piece of equipment.	Asks to specify identity information to the system using RFID or by specifying an identification code.
Scans the RFID-tag or inputs the identifier	Shows a screen similar to the screen showing the details of equipment, but with editable text fields where information about the piece of equipment can be added, including a field specifying the category the piece of equipment is in, which may or may not exist yet.
Fills out the form with text fields and specifies the category of the equipment. Then signals that all information has been added	Signals that the information has been added to the database.

Postcondition: The information about the piece of equipment has been added to the database, and has been linked to the unique identifier or RFID-tag.

9.2. **Update catalog information about equipment**
Actors: Technical workers.
Goal: To update the information in the database about a piece of equipment.
Preconditions: The piece of equipment is registered in the database.
Summary: The actor opens the detailed view of the item, and signals that he/she wants to edit the item. The actor then changes the information about that piece of equipment and saves these changes to

the database.
Related use cases: View details of equipment (8.2.)
Steps:

Actions taken by actor	System response
Opens a detailed view of a piece of equipment, according to use case "View details of equipment" (8.2.), then signals that he/she wants to change the information about that piece of equipment.	Makes the text fields with the details of the equipment editable, and shows an option where the identifier of the equipment can be changed by scanning an RFID-tag or specifying a unique identifier.
Changes the information about the piece of equipment where necessary, and signals to the system that he/she is done editing.	Signals that the information has been saved to the database.

Postcondition: The information about the piece of equipment has been updated in the database.

9.3. **Remove equipment from the catalog**
Actors: Technical workers.
Goal: To remove information about a piece of equipment from the database.
Preconditions: The equipment is currently registered in the database.
Summary: The actor opens the detailed view of the item, then signals that he/she wants to remove the item from the database. The item is then removed from the database.
Related use cases: View details of equipment (8.2.)
Steps:

Actions taken by actor	System response
Opens a detailed view of a piece of equipment, according to use case "View details of equipment" (8.2.), then signals that he/she wants to delete the information about that piece of equipment.	Asks for confirmation to delete the item.
Chooses to delete the information of the item	Confirms that the information about the piece of equipment has been deleted from the database.

Postcondition: The information about the piece of equipment has been deleted from the database.

10. Manage maintenance information

10.1. **View maintenance reports**
Actors: Technical workers.
Goal: To view a list of maintenance reports.
Preconditions: The home screen of the equipment maintenance application is open.
Summary: The actor views a list of maintance reports.
Related use cases:

Steps:

Actions taken by actor	System response
Specifies he/she wants to view a list of maintenance reports	Opens a view showing a list of maintenance reports

Postcondition: A view with a list of maintenance reports is shown.

10.2. **Add maintenance report**
Actors: Technical workers.
Goal: To add a maintenance report about a piece of equipment.
Preconditions: The equipment is registered in the system.
Summary: The actor specifies that he/she wants to add a maintenance report in the detail screen of an item or in the view containing a list of maintenance reports. A screen is opened in which information

can be specified. The information is then saved to the database when the actor is done.
Related use cases: View details of equipment (8.2.)
Identify a piece of equipment (7.)

Steps:

Actions taken by actor	System response
Option 1: Opens a detailed view of a piece of equipment, according to use case "View details of equipment" (8.2.), then signals that he/she wants to add a maintenance report about that item.	-
Option 2: Opens the list of maintainance reports, according to use case "View maintenance reports" (10.1.), then signals that he/she wants to add a maintenance report.	Shows a screen in which the user can select an item, according to use case "Identify a piece of equipment" (7.).
Specifies the identity information	-
-	Shows a view in which the actor can specify maintenance information and state of the particular piece of equipment, and a confirmation option.
Specifies the maintenance information and changes the state of the piece of equipment, if necessary. Then confirms when done.	Signals that the maintenance report has been added.

Postcondition: A new maintenance report has been added.

10.3. **View maintenance report**
Actors: Technical workers.
Goal: To view a maintenance report about a piece of equipment.
Preconditions: The equipment is registered in the system.
Summary: The actor specifies that he/she wants to view a maintenance report in the detail screen of a piece of equipment or in the view containing a list of maintenance reports. A screen is opened in which the maintenance report is shown.
Related use cases: View details of equipment (8.2.)
Identify a piece of equipment (7.)

Steps:

Actions taken by actor	System response
Option 1: Opens a detailed view of a piece of equipment, according to use case "View details of equipment" (8.2.), then signals that he/she wants to view the maintenance report(s) about that piece of equipment.	Shows a list of maintenance reports about the piece of equipment
Selects the maintenance report he/she wants to edit	-
Option 2: Opens the list of maintainance reports, according to use case "View maintenance reports" (10.1.), then opens a maintenance report shown in the list.	-
-	Shows a view in which the full maintenance report is shown.

Postcondition: A view with the the full maintenance report about an item is shown.

10.4. **Edit maintenance report**
Actors: Technical workers.

Goal: To update the information in a maintenance report of a piece of equipment.
Preconditions: The maintenance report matches an item in the database.
Summary: The actor opens the maintenance report, then signals that he/she wants to edit the report. The actor changes the information in the maintenance report and saves the information to the database.
Related use cases: View maintenance report (10.3.)
Steps:

Actions taken by actor	System response
Opens the maintenance report he/she wants to edit, according to use case "View maintenance report" (10.3.), then signals that he/she wants to edit that report.	Makes all information in the maintenance report editable.
Edits the information in the maintenance report and signals that he/she is done.	Confirms that the edited maintenance report has been saved to the database.

Postcondition: The edited maintenance report has been saved to the database.

4 User Interface

In this section, we present several storyboards to show how the actors will interact with our system through the graphical user interface (GUI). The storyboards show the functionality of a particular subsystem, such as the iPhone, iPad or RFID-system, and are presented in the appropriate subsections below.

4.1 Storyboards iPhone

4.1.1 Map Screen

Figure 4 is the main screen of the iPhone application. On this screen the user is able to see a map (1) of the disaster area, in particular the work site in which the rescue team is working in. There are different symbols located on the map, representing a victim (3) or a danger (4) or his/her current location. On the bottom of the screen there is a tool bar (2) which allows the user to preform certain tasks; he/she is able to centralize the map on his/her current location by selecting symbol 5 from the tool bar. The user is able to adding symbols to the map by selecting the symbols (3 or 4) on the tool bar. When adding a symbol to the map the Detailed Information Screen will appear with blank sections for the user to fill in. If a user wants to view detailed information about the symbol, he/she will select the corresponding symbol and the Detailed Information Screen will appear with the detailed information.

4.1.2 Detailed Information Screen

Figure 5 is the Detailed Information screen of the iPhone application. In this screen the user is able to add detailed information about a symbol (victim or danger). On the very top of the screen is a menu bar (1) where the user is able to press back an return to the Map Screen. He is also able to delete the symbol or edit the detailed information by selecting the corresponding buttons on the menu tool bar. In the main display (2) is where all the information can be found and placed. It is divided into sections, like symbol (where you are able to edit the symbol), status of the rescue, remarks and photos. The user is able to see a photo by selecting the thumbnail of the photo and the Photo Screen will be displayed. A user can also add photos by selecting the add photo button (3). By doing this the camera of the iPhone will be activated and the user will be able to take a photo. Once a photo has been taken the photo will be displayed in the Photo Screen.

4.1.3 Photo Screen

In Figure 6 the user is able to see the taken/selected photo. Like the Detailed Information Screen there is a menu tool bar (1), where by pressing back the user will be brought back to the Detailed Information Screen, delete will delete the photo. There is also filter button, on a photo the user is allowed to make annotations in the form of symbols or drawings (of circles and arrows). By selecting filter it will make

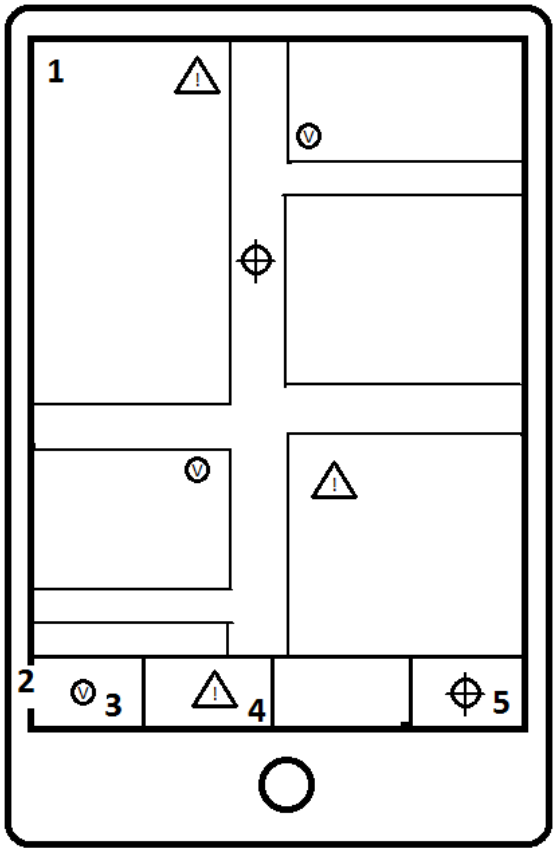

Figure 4: The map screen of the iPhone.

all the annotations invisible. To edit drawing object already made the user presses the button edit on the menu tool bar (1). He is also able to delete the drawn object by pressing delete. To add a drawing object to a photo the user needs to press either the circle symbol (5) or the arrow symbol (6) which is located at the bottom of the screen in the tool bar (2). The user is also able to add symbols to the photo by pressing the add victim (3) or danger (4) button in the tool bar (2). Once the user has indicated where on the photo the symbol should be placed the Photo Detail Screen will appear with blank sections to be filled in. If the user wants to view the details of a symbol he will select it and the Photo Detail Screen will appear.

4.1.4 Photo Detail Screen

Figure 7 shows the screen of detailed information of a symbol placed on a photo. This is very similar to the Detailed Information Screen, where there is a menu tool bar (1) with back button (which brings the user back to the Photo Screen), delete button (which deletes the symbol) and edit button. There is also a display section how ever it is limited compared to the Detailed Information Screen with only a symbol (2) and remark (3) section.

4.2 Storyboards iPad

4.2.1 Main menu

Figure 8 shows the initial screen of the iPad application. This screen only contains three buttons. Pressing the view map button, indicated by number one, will take the user to the map screen. By

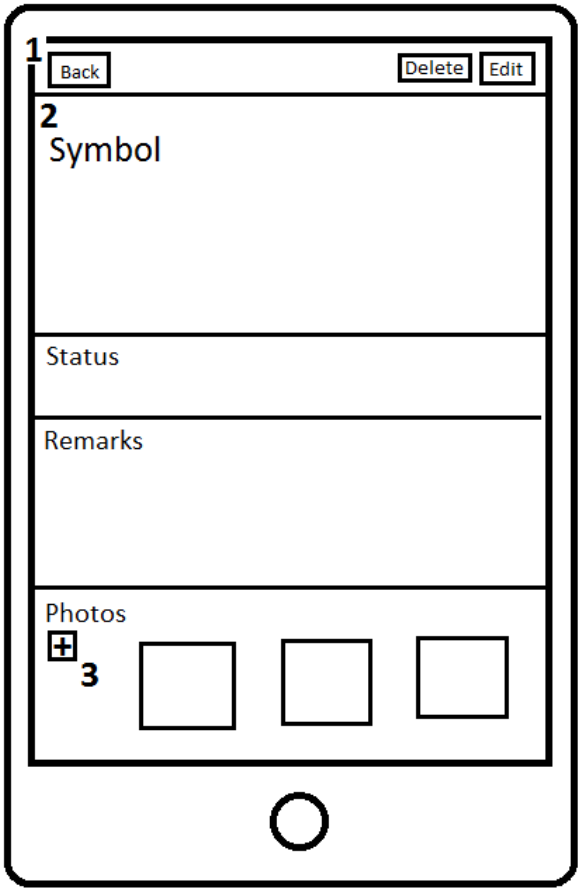

Figure 5: The detailed information screen of a symbol.

pressing the second button, statistics about every team can be seen in a table. Button number three will show an overview of all equipment and it's status.

4.2.2 Map screen

Figure 9 shows the location of the teams and some information about their work site. Pressing the button indicated by number one, will center the map on the position of team Alpha. Button two switch the user to the team statistics screen. Button three, the back button, will return the user to the menu screen.

The map itself is similar to the map used by the iPhone application. All symbols that can be seen in that application, can also be seen here, except for the symbol indicated by number four. This symbol is unique to the iPad and indicates the location of the commander of a search and rescue team.

4.2.3 Team Statistics

Figure 10 shows the screen with the statistics of every team in the form of a table. The back button, indicated by number on, will return the user to the main menu. Number two represents the top row of the table, showing the user which team's statistics can be seen in which column. The first column (3), explains which information can be seen on every row. Finally, by pressing one of the buttons at the bottom of the page (4), the application will switch to the map screen on center on the gps-position of the commander of the resue team.

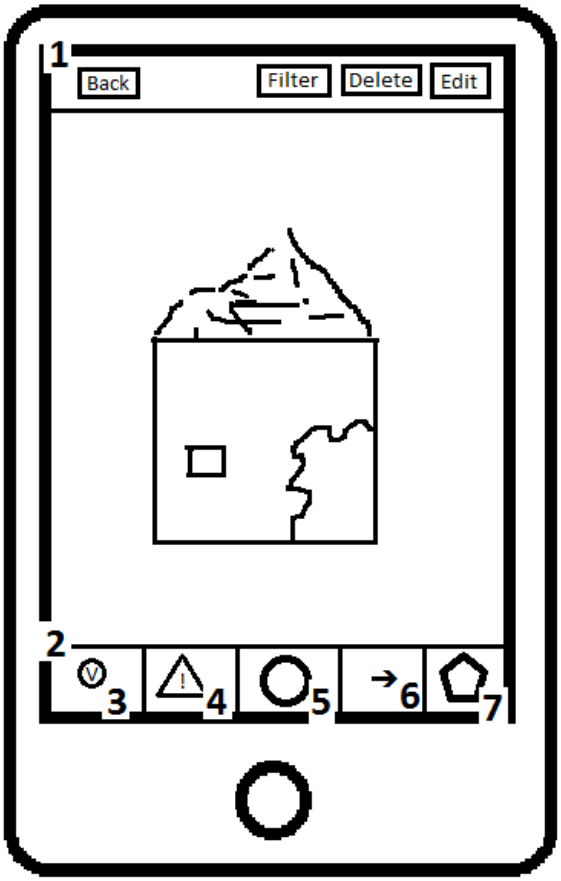

Figure 6: The photo screen.

4.2.4 Equipment Statistics

Finally, figure 11 shows the a list of all equipment. The back button (1), once again returns the user to the main menu. On the left side of the screen, at number three, all equipment is listed. By pressing an item, details about this piece of equipment will be shown on the right (5). Button four, Add report, allows the user to add a maintenance report to the selected item.

The view reports button, indicated by number two, will change the list (3) into a list of maintenance reports. This will also change the contents of the right side of the screen. Section five will show a summary of the maintenance report instead of equipment status. The button at the top right of the screen will change to allow the user to edit a report, instead of adding one.

4.3 Storyboards RFID-system

Figure 12 shows the start screen for the RFID-application. No objects are selected yet.

4.3.1 Checking out equipment

When a search and rescue team member wants to check out the items that is taken to a work site, he/she selects a button from the buttons 1-4 that corresponds to the team the team member is in. Figure 13 shows that button 2 has been selected, to check out equipment for team B.

The objects can now be checked out for team B. The RFID-tags of the objects can be scanned one at a time. The objects scanned then appear in list 9, and a counter is kept and shown in 8, as shown in

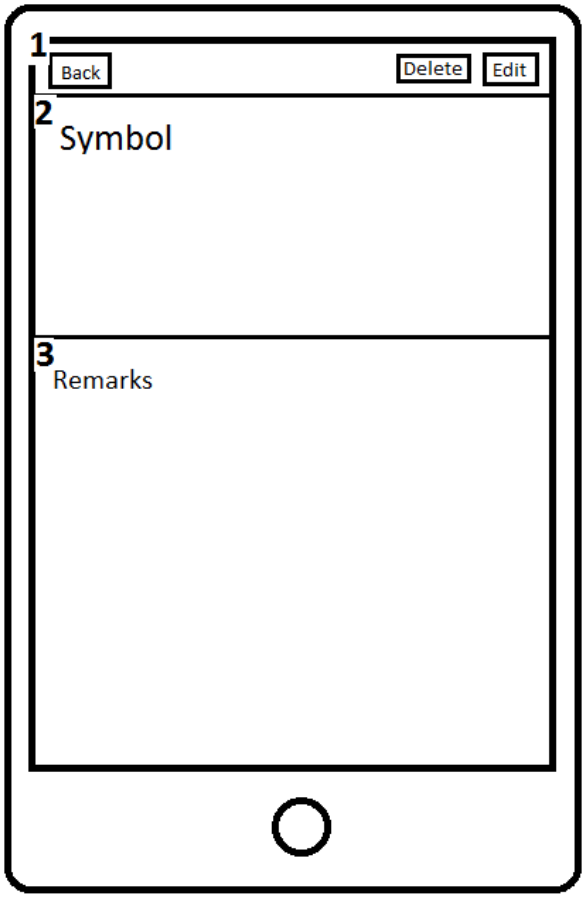

Figure 7: The detailed information screen of a symbol on a photo.

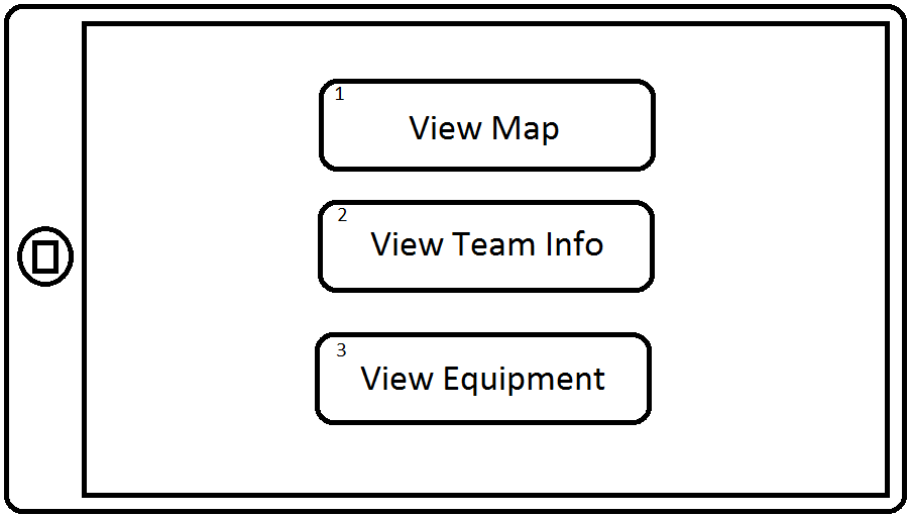

Figure 8: The start screen of the iPad application.

Figure 14. The list can be scrolled to view all equipment, if the list becomes too big to fit in the screen. When the team member is done scanning the equipment, he verifies that all objects have been scanned.

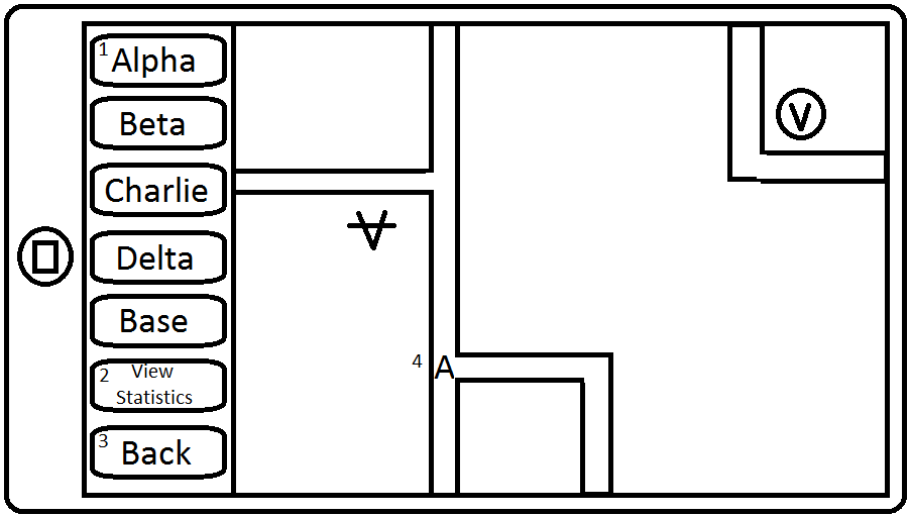

Figure 9: The map of each team's work site.

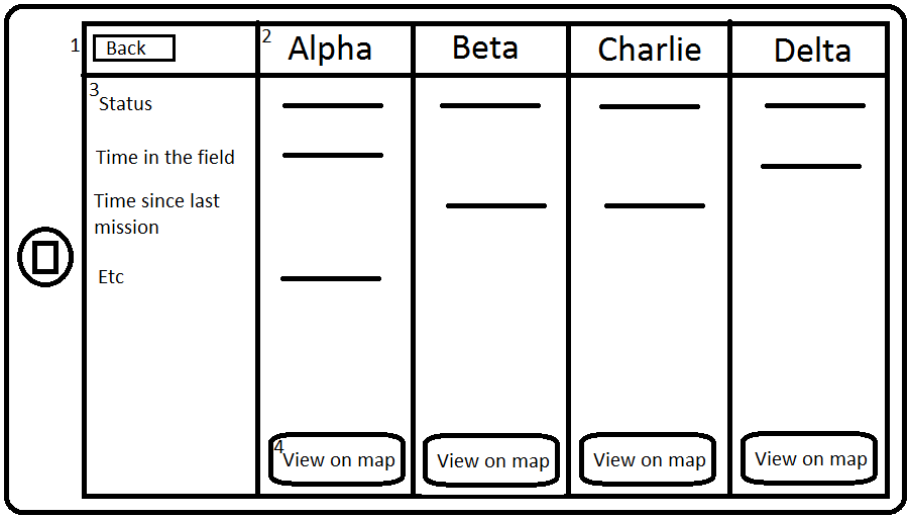

Figure 10: A table showing important information about the rescue teams.

If there are any object he has not scanned, he scans that equipment. If some equipment has been scanned wrongly, these objects can be removed by pressing a button similar button 10 next to the equipment that needs to be removed. If all objects are scanned correctly, button 7 is pushed. Then, the screen is returned to the start screen of the RFID-system, as shown in Figure 12.

4.3.2 Checking in equipment

When a search and rescue team member gets back to the base camp, the equipment is checked out again. To start checking out the equipment, button 5 is selected in the start screen, as shown in Figure 15.

Then, the objects can be checked in. The RFID-tags of the objects can be scanned one at a time. The objects scanned then appear in list 9, and a counter is kept and shown in 8, as shown in Figure 16. The list can be scrolled to view all equipment, if the list becomes too big to fit in the screen.

When the team member is done scanning the equipment, he verifies that all objects have been scanned. If there are any object he has not scanned, he scans that equipment. If some equipment has been scanned wrongly, these objects can be removed by pressing a button similar button 10 next to the equipment that needs to be removed. If all objects are scanned correctly, button 7 is pushed. Then, the screen is returned to the start screen of the RFID-system, as shown in Figure 12.

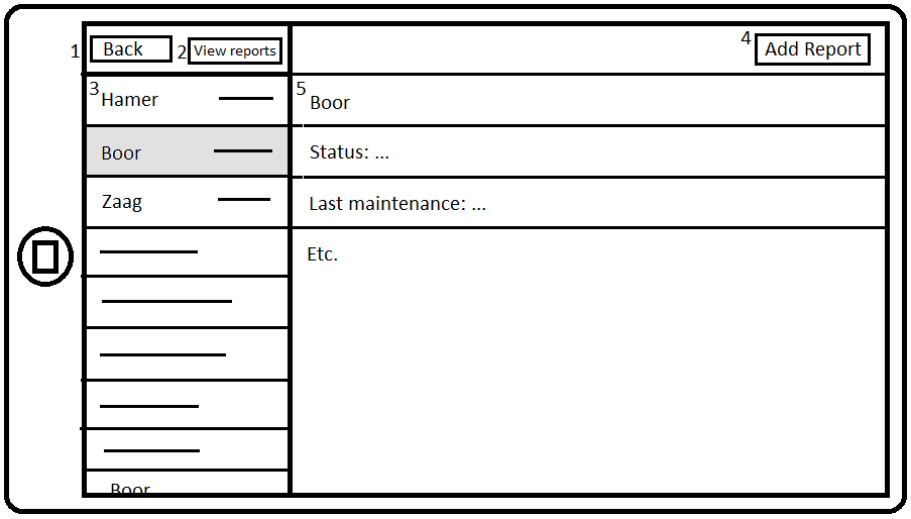

Figure 11: A list of all equipment and their details.

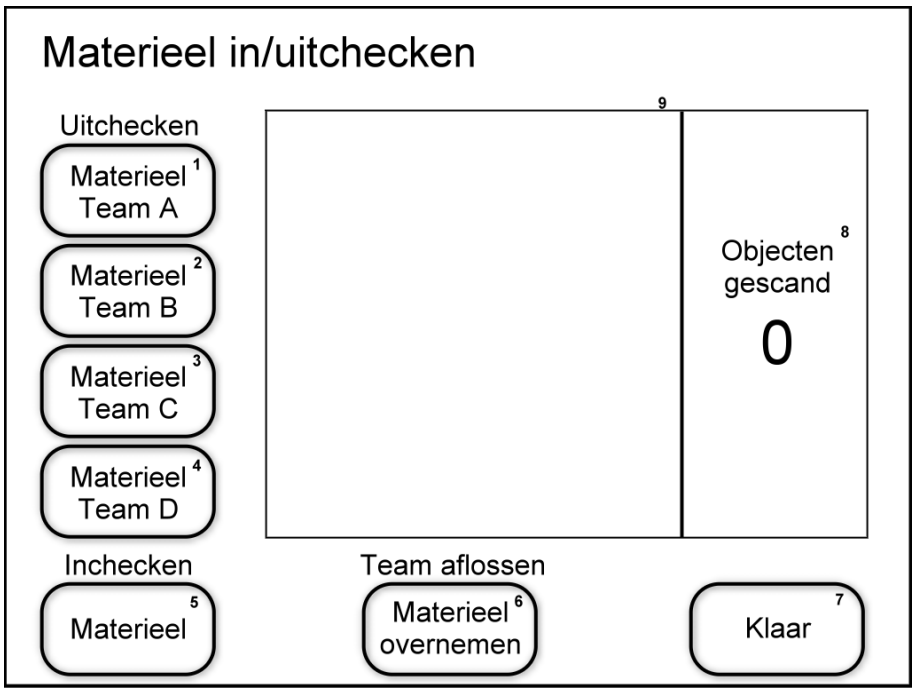

Figure 12: The start screen of the RFID-application.

4.3.3 Taking over equipment from another team

When a search and rescue team is relieving another team, the team wants to take over the equipment from the other team. A member of the relieving team then selects button 6. Figure 17 shows the screen in which the team member can specify which team they are taking over equipment from.

Then, the team member from the relieving team selects one of the options 1-4, corresponding to the team he or she is in. Figure 18 shows the screen after option 1 (Team A) has been selected as the relieving team.

Then, one of the options 5-8 is selected, corresponding to the team that is being relieved, and from whom the equipment is taken over. Figure 19 shows the screen after option 6 (Team B) has been selected as the team that is being relieved.

Then, the transfer of equipment is confirmed by selecting option 9, after which the screen is returned

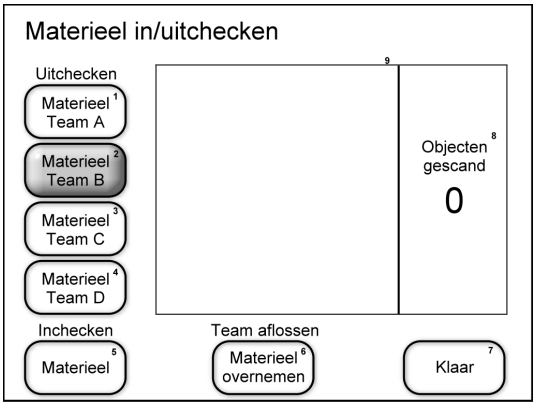

Figure 13: The start screen of the RFID-application, in which option 2 is selected.

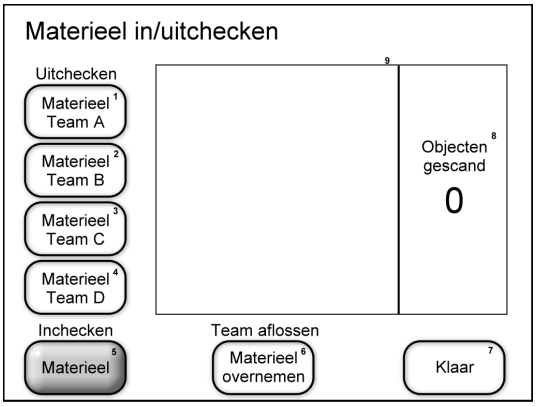

Figure 15: The start screen of the RFID-application, in which option 5 is selected.

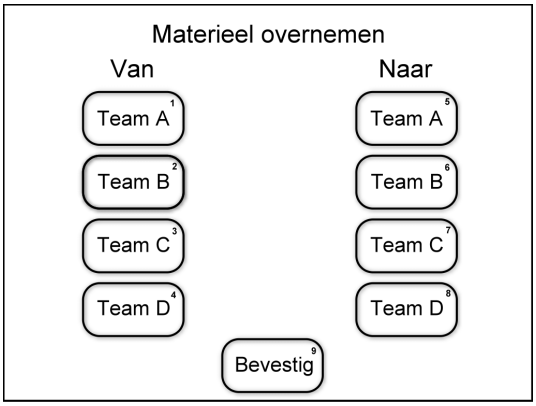

Figure 17: The screen of the RFID-application in which a team can take over equipment from another team.

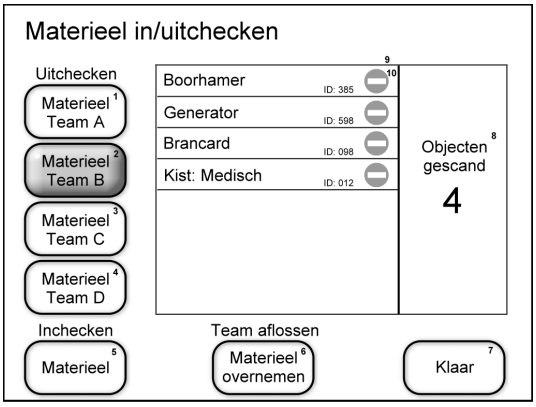

Figure 14: The start screen of the RFID-application, in which option 2 is selected and objects have been scanned, which are shown in list 9.

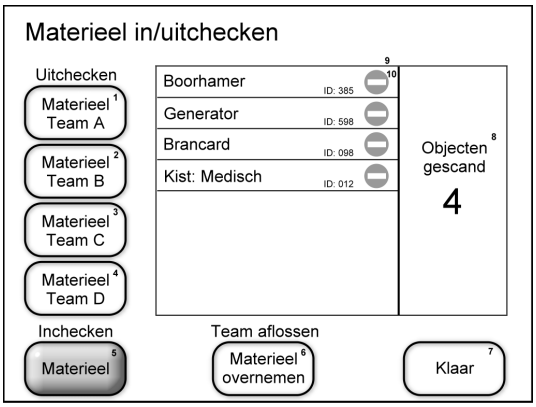

Figure 16: The start screen of the RFID-application, in which option 5 is selected and objects have been scanned, which are shown in list 9.

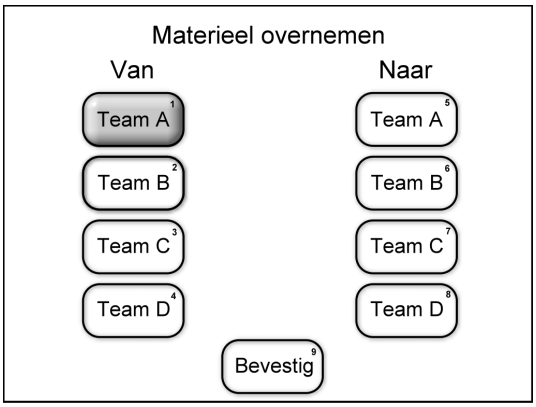

Figure 18: The screen of the RFID-application in which a team can take over equipment from another team. Team A (option 1) has been selected as the relieving team.

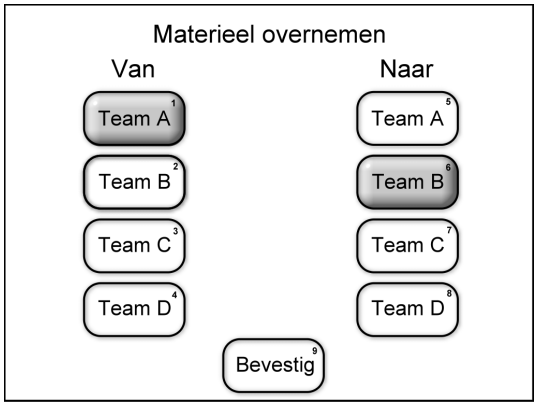

Figure 19: The screen of the RFID-application in which a team can take over equipment from another team. Team A (option 1) has been selected as the relieving team, Team B (option 6) has been selected as the team that is being relieved.

to the start screen of the RFID-system, as shown in Figure 12.

References

[2] K. Chung. Low Cost and Reliable RFID Tags for All Frequencies. 2003. URL `http://www.avantetech.com/uploads/pdf/ReliableRFID.pdf`.

[3] T. de Greef, A. Oomes, and M. Neerincx. Distilling Support Opportunities to Improve Urban Search and Rescue Missions. *Human-Computer Interaction. Interacting in Various Application Domains*, pages 703–712, 2009.

[4] Ir. M. de Hoogh. personal communication, 2010.

[] M.A. Neerincx. Situated cognitive engineering and its design rationale. 2010.

[5] Y. Pang. personal communication, 2010.

[6] C.M. Roberts. Radio frequency identification (rfid). *Computers & Security*, 25(1):18 – 26, 2006. URL `http://www.sciencedirect.com/science/article/B6V8G-4J61650-1/2/eaf8f6227274fe012c748146e1585a20`.

Appendix D

Design Document

1 Introduction

In the Design Document we look at the architect of our system. In Section 2 is a diagram real world classes. Then in section 3 we have the state diagrams for the iPhone, iPad and RFID based on the user interface. Following that in section 4 we have the technical design, with diagrams of the model, controllers and communication package.

2 Real World Class Diagram

Figure 1 shows a diagram showing the real-world objects, of which representations will be stored in our database.

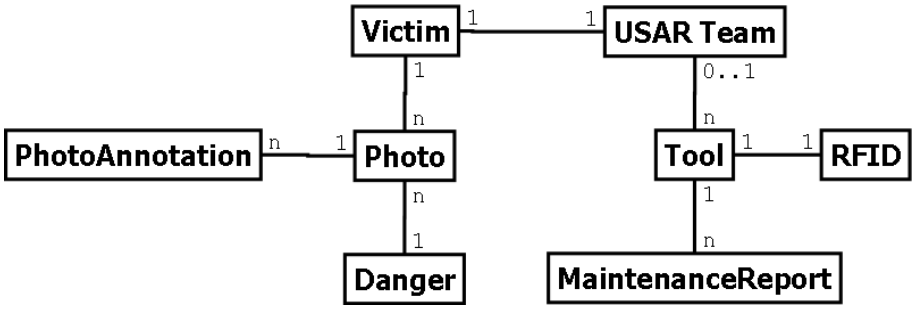

Figure 1: The "Real World" class diagram.

3 State Diagrams

In this section the state diagrams will be shown for each applications, iPhone, iPad and RFID. These state diagrams were derived from our user interface, to be able to see what type of states the system could be in during exicution. By creating these state diagrams we are able to determine what possible state transitions are possible and the different paths one could follow.

1. **iPhone Sate Diagram**

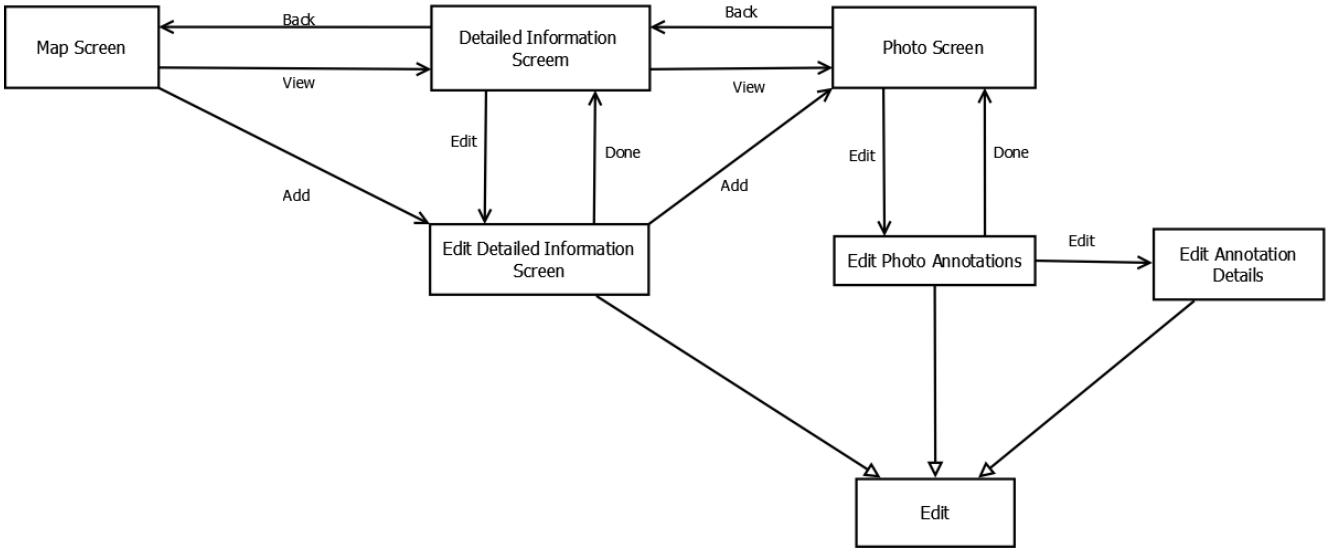

Figure 2: The iPhone State Diagram.

2. **iPad State Diagram**

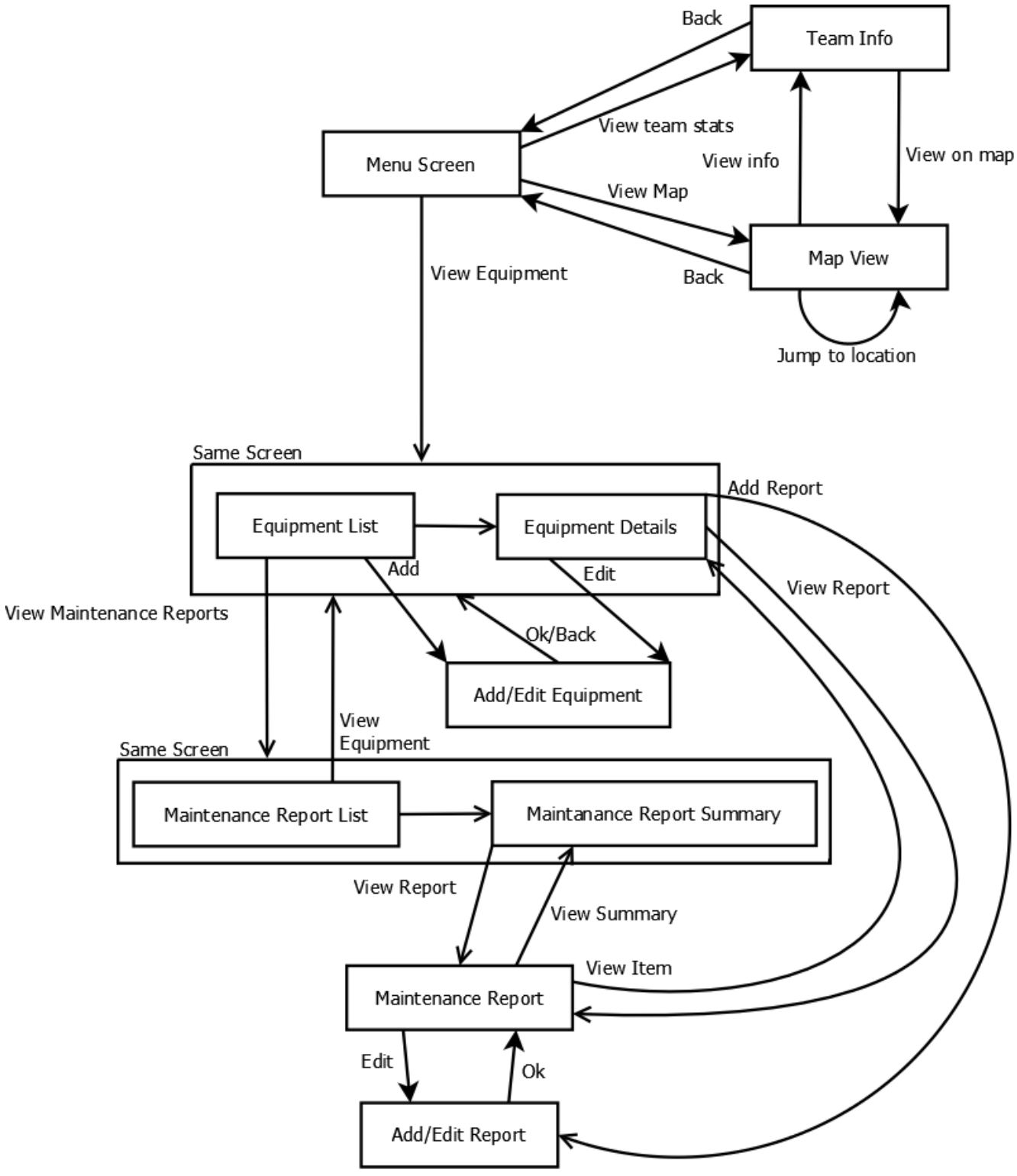

Figure 3: The iPad State Diagram.

3. **RFID State Diagram**

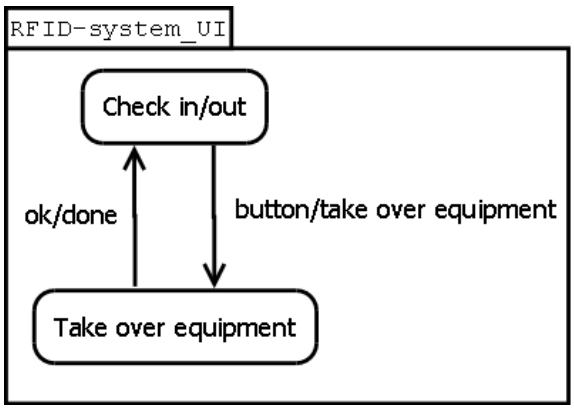

Figure 4: The RFID State Diagram.

4 Technical Design

This section contains the class diagrams of our system. The paradigm we have applied is Model-View-Controller. Each application has its own view (the interface) and controller. The model is shared between all applications. Figure 5 shows the package structure. We will provide class diagrams for the Model, the Controllers for the iPad, iPhone, and RFID-system, and the Communication package. For the Server we will not make a class diagram, since we intend to use the server as it is, with very little additions. Also, we will not make a class diagram for the views, since these will be made using the Interface Builder provided by the iPhone SDK for the iPhone and iPad views, and the interface builder for .NET-applications for the RFID-system views.

Packages Within our program we defined different packages to enhance the re usability of components in our program. There are a total of 9 packages that we define, we created a view package for each application (iPhoneView, iPadView, RFIDView) and also a controller package for each application (iPhoneController, iPadController, RFIDController). There are also a Model and Communication package. In the View packages are classes that have to do with the user interface, what the user is able to see and interact with. Then there is the Model package, this contains the actual objects that store data and. The controller package contains classes that connect the models with the view or user interface. There is also a communication package that contains a client class, this class takes care of the communication between the our system and a server.

1. **Package Diagram**

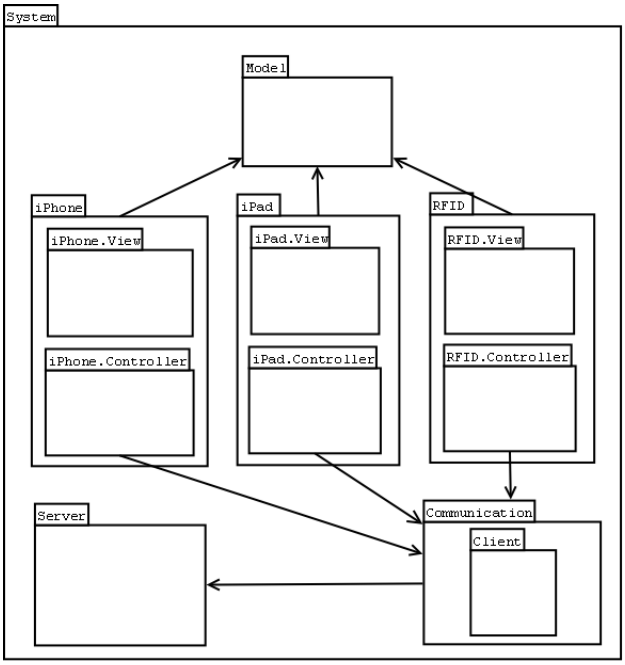

Figure 5: The package diagram.

2. **Model** The Model contains all classes that hold information. The Model is shared between the iPhone, iPad, and RFID-application.

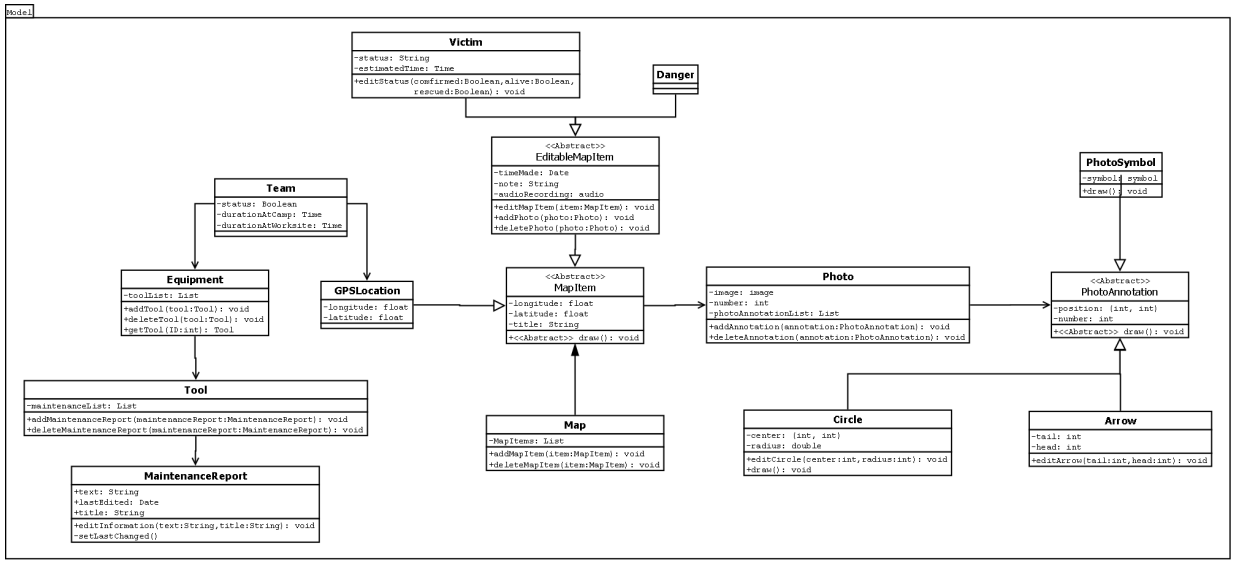

Figure 6: The Model.

3. **iPad Controller** The Controller package of the iPad application contains all classes that are used to respond to the user actions, by providing methods to perform the required operations.

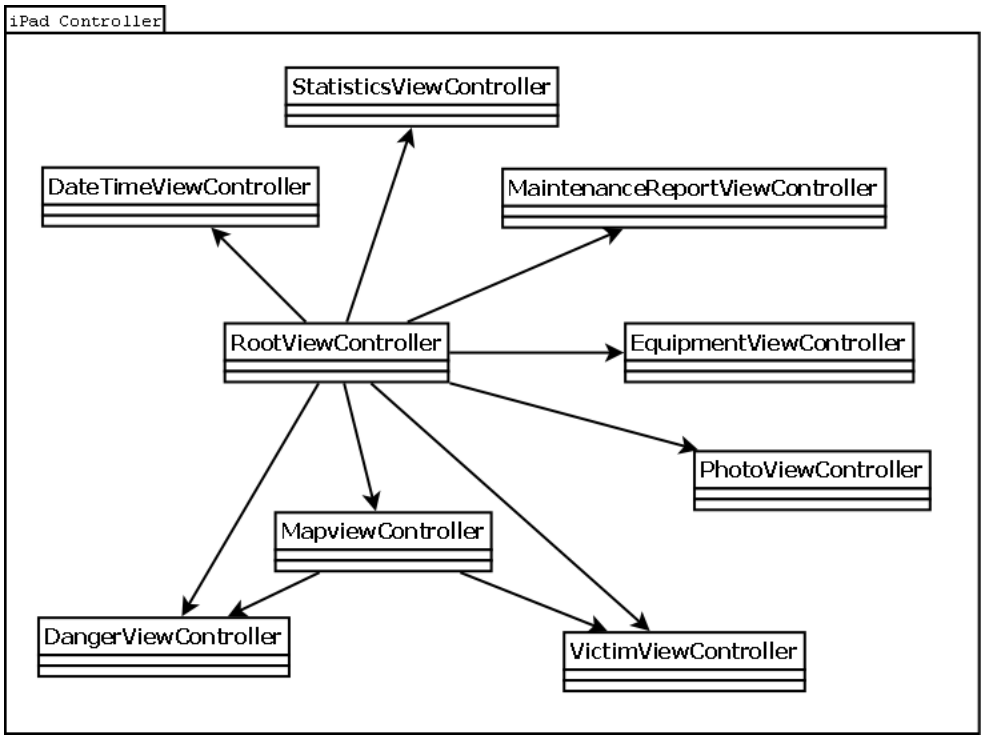

Figure 7: The Controller for the iPad application.

4. **iPhone Controller** The Controller package of the iPhone application contains all classes that are used to respond to the user actions, by providing methods to perform the required operations.

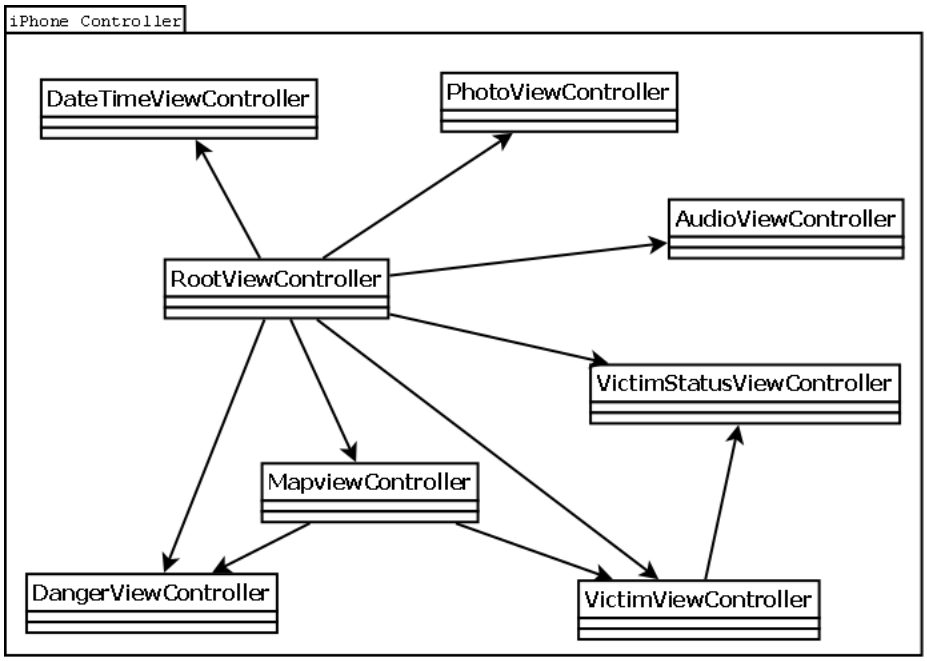

Figure 8: The Controller for the iPhone application.

5. **RFID Controller** The Controller package of the RFID application contains all classes that are used to respond to the user actions, by providing methods to perform the required operations. These operations also include communication with the RFID-reader, either through a library or by communicating with a separate program.

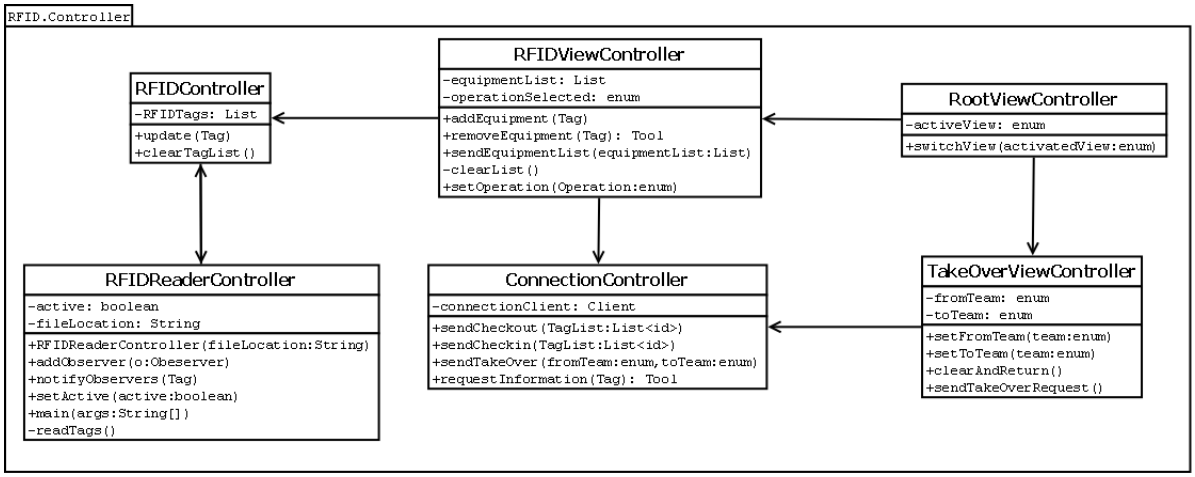

Figure 9: The Controller for the RFID application.

6. **Communication** The Communication package is used by the iPhone-, iPad-, and RFID-application to communicate with the server. The package contains a single class, with methods that can be called to send specific data to the server, or request data from the server.

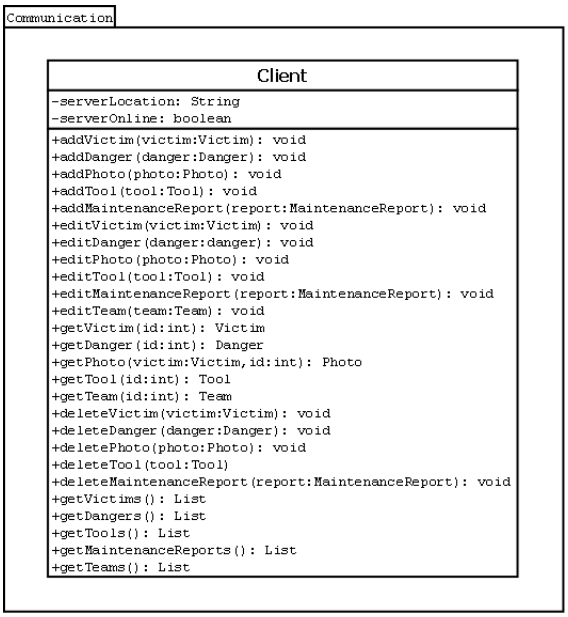

Figure 10: The Communication package.

Appendix E

Test and Implementation Plan

1 Implementation Planning

In this chapter, we describe the planning and phases in which our system will be developed. First, we describe the priorities set for the different parts of our system in a MoSCoW-document. Then, we give a planning in which we specify the activities for each day and milestones that specify the products delivered.

We will impelement our system using an agile approach, by first implementing the most important functionality and creating a working prototype. Then, we extend the functionality of our program incrementally. The resulting programme can then be tested, in order to minimise the risk of not having a working program at the end of this project.

1.1 MoSCoW

In this section we present the priorities we have set for each type of functionality of our system. These priorities are presented in the form of a MoSCoW-document. The functionality mentioned under *Must have* will have to be implemented for the system to be useful, and, consequently, will be implemented first. Second, the functionality mentioned under *Should have* is implemented, since this functionality will improve the usability of our system greatly. Functionality mentioned under *Could have* are implemented optionally, if there is any time left after implementing and testing the functionality for Must have and Should have, since the consequences for not having this functionality do not impact system performance much. The functionality mentioned under *Would like to have* will not be implemented during this project, but may be implemented in future projects as an extension to this project.

Must have

- iPhone / iPad: View and navigate map.
- iPhone / iPad: View dangers and victims on a map.
- iPhone: Current location can be viewed on a map.
- iPad: Location of different teams can be viewed on a map.
- iPhone: Victim status, notes and photos can be added, edited, and viewed for each victim on a map.
- iPhone: Notes and photos can be added, edited, and viewed for each danger on the map.
- RFID: Checking equipment in and out.
- iPad: Location of equipment can be viewed.
- iPad: A list of equipment can be viewed.

Should have

- iPhone: Annotations can be added to photos.
- iPhone: Sound recordings can be added to victims and dangers.
- iPad: The status of equipment can be changed and viewed.
- iPad: Team statistics can be viewed.
- RFID: Equipment can be transferred to another team.
- RFID: A warning is displayed when trying to check out a defective piece of equipment.
- iPad: A list of defective equipment can be viewed.

Could have

- iPhone: Gesture recognition for simple gestures.
- iPad: Situational reports can be viewed.
- iPad: Maintenance reports can be viewed, added and edited.
- iPad: Problems with equipment can be viewed.
- iPad: The list of equipment can be sorted and filtered by usage since last maintenance.

Would like to have

- iPhone: INSARAG-symbol recognition with connection to spray can.
- RFID: Integration with iPad application.
- RFID: Checking in and out automatically (e.g. by passing through a gate).
- General: Full integration with surface table application.
- iPad: View a timeline similar to the timeline of the surface table application.

1.2 Planning

Appendix A contains an updated version of the planning, showing when implementation of the must haves and should haves will be finished.

2 Testplan

2.1 Unit Tests

Because of the limited timeframe and the difficulty of creating unit tests for interface classes, these will only be used when necessary. Most parts of our system will be tested using other testing methods described in the next couple of sections. Since both the iPhone and iPad applications will be written in objective-C, OCUnit-tests will be made when necessary for methods in the model and controller. The RFID and server applications, written in C#, will be tested using NUnit-tests.

2.2 Scenario Testing

Often, unit tests can not properly test the interaction between different classes. For some methods, it will not even be possible to create unit tests. Therefore, we will also test using scenario testing for the user interface. To do this, we will manually execute all use cases, as described in section 4.5 of the requirements analysis document. If all these use cases can be executed without failure, scenario based testing will be finished.

2.3 State-based Testing

Section three of the design document contains several state diagrams of the three different applications. Another test method we will use is based on these state diagrams. We will test whether the transitions from one state to another are executed properly by applying the appropriate events to traverse the a path in the diagram. If all changes are applied correctly and the state has changed properly, the test has passed.

2.4 Requirements Evaluation

To test whether the system actually meets our requirements, another test method will be used if possible, based on the claims in section 4.2 of the requirements analysis document. We will attempt to test our system by allowing USAR.nl members to interact with it. However, there is a good chance it will not be possible to schedule this, mainly due to the busy schedule of the rescue team members.

Planning IN3405 Bachelorproject

	Legend			
	PvA	Plan van Aanpak		
	OV	Oriëntatieverslag		
	RAD	Requirements Analysis Document		
	ADD	Architectural Design Document		
	TDD	Technical Design Document		
	TIP	Test & Implementatieplan		
Weeknr	**Date**	**Action**	**Deadline**	**Deliverable**
1	4/19/10	Make Planning / Start Weblog		
	4/20/10	PvA		
	4/21/10	PvA		
	4/22/10	PvA / OV	PvA finished	PvA
	4/23/10	OV	Concept OV finished	
	4/24/10			
	4/25/10			
2	4/26/10	RAD/Prepare for brainstorm-session		
	4/27/10	RAD/Prepare for brainstorm-session		
	4/28/10	Brainstorm-session		
	4/29/10	OV / RAD	OV finished	OV
	4/30/10	Koninginnedag		
	5/1/10			
	5/2/10			
3	5/3/10	RAD		
	5/4/10	RAD		
	5/5/10			
	5/6/10	RAD		
	5/7/10	RAD		
	5/8/10			
	5/9/10			
4	5/10/10	RAD / ADD	12:00 RAD finished	RAD
	5/11/10	ADD		
	5/12/10	ADD		
	5/13/10	Hemelvaartsdag		
	5/14/10	ADD	17:30 ADD finished	ADD
	5/15/10			
	5/16/10			
5	5/17/10	TDD		
	5/18/10	TDD	17:30 TDD finished	TDD
	5/19/10	TIP		
	5/20/10	TIP	17:30 TIP finished	TIP
	5/21/10			
	5/22/10			
	5/23/10			
6	5/24/10	2e Pinksterdag		
	5/25/10	iPhone/iPad GUI & RFID Must Have		
	5/26/10	iPhone/iPad GUI & RFID Must Have		
	5/27/10	iPhone/iPad GUI & RFID Must Have		
	5/28/10	iPhone/iPad GUI & RFID Must Have		
	5/29/10			
	5/30/10			

Weeknr	Date	Action	Deadline	Deliverable
7	5/31/10	iPhone/iPad GUI & RFID Must Have	GUI for iPad and iPhone	GUI
	6/1/10	Conference WebandBeyond		
	6/2/10	Must Have		
	6/3/10	Must Have		
	6/4/10	Must Have		
	6/5/10			
	6/6/10			
8	6/7/10	Must Have\Should Have		
	6/8/10	Must Have\Should Have		
	6/9/10	Must Have\Should Have		
	6/10/10	Must Have\Should Have		
	6/11/10	Must Have\Should Have		
	6/12/10			
	6/13/10			
9	6/14/10	Should Have	17:30 First runnable finished	First runnable
	6/15/10	Should Have		
	6/16/10	Should Have		
	6/17/10	Should Have		
	6/18/10	Should Have		
	6/19/10			
	6/20/10			
10	6/21/10	Bugfixing / Final Report	09:00 No New Functionality	
	6/22/10	Bugfixing / Final Report		
	6/23/10	Bugfixing / Final Report		
	6/24/10	Bugfixing / Final Report		
	6/25/10	Bugfixing / Final Report	17:30 Final Report finished	Final Report
	6/26/10			
	6/27/10			
11	6/28/10	Bugfixing		
	6/29/10	Bugfixing		
	6/30/10	Bugfixing		
	7/1/10	Bugfixing		
	7/2/10	Bugfixing	Final Program version	Program
	7/3/10			
	7/4/10			
12	7/5/10	Presentation		
	7/6/10	Presentation		
	7/7/10	Presentation	17:30 Presentation finished	
	7/8/10	Presentation		
	7/9/10	Presentation	Final DEADLINE Presentation!!!	

Appendix F

Gesture Recognition

As mentioned above for the iPhone it might be possible to use gesture recognition to recognize symbols that a rescue worker spray paints on a wall. We have done some research to see how far this is possible.

The iPhone has an accelerometer, which is able to measure the acceleration on the three axes (X,Y,Z). With this information it is possible to calculate the direction the iPhone is moving.

During our research we were able to have an interview with Ir. M. de Hoogh [5] who works at the Technical University of Delft. For the past half year he has been working on a similar program for the wii-mote which is able to recognize numbers. We based our research on his program, because even though it was made for the wii-mote, the input and output, and concepts are the same for the iPhone. The program uses a Hidden Markov Model to be able to recognize gestures.

Method The first step is to create samples to train the program to recognize the gesture. Multiple sequences of the gesture are recorded, grouped and categorized by the sample length (to allow quicker recognition, will only compare to sequences of same length). These data points are then associated to a point in a 3D space. Once all the sequences are placed in the 3D space, clusters form where the centre of the cluster will be represented as a reference point. Once a gesture is made, to be recognized, based on the on the distance between the data point (of the input gesture) and the reference point (centre point of the training samples) a probability value can be calculated. With this probability the program will be able to make a choice, of which gesture is being performed.

Problems encountered There were some of the problems that became apparent. For symbols that can be drawn in a single flowing motion, the program is very sensitive to the speed of which the symbol is being drawn, the angle of which the iPhone is being held while the symbol is being drawn and the size of the symbol that is being drawn. These are all personal traits that are different from person to person. It is not possible for person A to train the iPhine to recognize symbols and then have it recognize person B's symbols. Even then it is hard to keep the same speed and size each time one draws a symbol let alone when there are days, weeks or even months in between.

Other problems that arise are that the symbols used in the USAR domain aren't always possible to draw in one flowing motion (e.g. an arrow). This means there either should be an on and off switch for tracking or not, both options have its own problems.

If we do not create an off on switch, the tracking will be constantly on during the drawing of the symbol. This is like drawing a symbol without taking the pen off the paper. Everyone

draws symbols differently and with these extra lines (connecting symbols/lines together) can lead to exponential amount of possibilities.

If we do decide to create an on and off switch, the problem is that we only get the raw data of acceleration. There is no orientation between the line segments.

Conclusion of Research Gesture recognition is a very interesting way of placing input into our system, however it is a very big task requiring a lot research and work. Due to its complexities and time constraints it won't be included in the scope of the project.

Having said this, with all the problems mentioned above doesn't mean that this isn't a possible or viable method of input. There are some possible solutions to the afore mentioned problems that could be considered. For example, when the tracking of movement is "off", it is possible to track the movement of the iPhine to get some type of orientation. An iPhone could be given to rescue workers to train and use it for gesture recognition.

There still needs to be more research done and questioned answered. For example, is it practical to try to recognize a gesture if you need to put information by hand anyway? A benefit vs cost analysis could also be preformed; how much extra effort will it take for it to recognize the right gesture? Should it recognize the wrong gesture how much effort is needed to correct this (how often could this occur)? The use of gesture recognition also restricts the rescue workers in how they draw their symbols (what order, speed, angle and size) is this a hindrance to the rescue workers or are they willing to adapt? This are just some of the things that need to be looked into.

Bibliography

[1] Web and beyond 2010). URL http://www.thewebandbeyond.nl/2010/website/.

[2] Dubai verslag. TNO internal Memo, 2009.

[3] K. Chung. Low Cost and Reliable RFID Tags for All Frequencies. 2003. URL http://www.avantetech.com/uploads/pdf/ReliableRFID.pdf.

[4] T. de Greef, A. Oomes, and M. Neerincx. Distilling Support Opportunities to Improve Urban Search and Rescue Missions. *Human-Computer Interaction. Interacting in Various Application Domains*, pages 703–712, 2009.

[5] Ir. M. de Hoogh. personal communication, 2010.

[6] M. Grootjen, L. Cdr, M.A.N.P. Dr, N.T. de Vries Capt, and Ghijben. Applying Situated Cognitive Engineering For Platform Automation in the 21st Century. 2009.

[7] M.A. Neerincx. Situated cognitive engineering and its design rationale. 2010.

[8] MA Neerincx, J. Lindenberg, N. Smets, A. Bos, L. Breebaart, T. Grant, A. Olmedo-Soler, U. Brauer, and M. Wolff. The mission execution crew assistant: Improving human-machine team resilience for long duration missions. In *Proceedings of the 59th International Astronautical Congress 2008*, 2008.

[9] Y. Pang. personal communication, 2010.

[10] C.M. Roberts. Radio frequency identification (rfid). *Computers & Security*, 25(1):18 – 26, 2006. URL http://www.sciencedirect.com/science/article/B6V8G-4J61650-1/2/eaf8f6227274fe012c748146e1585a20.