# GIS as a mobile tool for emergency management in practice

*Primarily to be used at events of extreme rain*


Azad Palmqvist, Bjarke Foss,
Tina Endersen & Zijad Cosic

MTM – Master of Geoinformation Management
4th semester, 2013
Aalborg University

Illustration on the front page: MTM group 4

**Title**

GIS as a mobile tool for emergency management in practice

**Study at Aalborg University**

MTM –
Master of Geoinformation Management

**Project period**

1st of September 2013 until 9th of January 2014

**Project group**

MTM-project group 4

**Project participants:**


……..
Azad Palmqvist,

……..
Bjarke Foss,

……..
Tina Endersen,

……..
Zijad Cosic


**Project supervisor**
Henning Sten Hansen,
Aalborg University - Copenhagen.

**Reports in circulation:** 7 reports.
**Pages:** 115 pages.
**Number of appendixes:** 13 attached appendices of a total of 29 pages.

Abstract


This report is about the making of a prototype of a mobile GIS tool for the combined Danish Emergency Rescue Service. This report aims to show the considerations to be taken in order to make a useful prototype. From user requirements, user interface design, and planned use to evaluation of theories and methods for implementing a practical technical solution. The prototype consists of a web application designed for tablets, and data from databases providing information published by and accessed through a web server.



Institute of Planning
Aalborg University

9th of July 2014

# Preface

## About this report

This report has been prepared during our 4th semester of the master's program in Geoinformation Management at Aalborg University from the 1st of September 2013 until the 9th of January 2014. Our study group consist of 4 students with a common interest in Geographic Information Systems (GIS) and tackling climate change. In this 4th semester project we will address the design of a geographical information tool to support preparedness and information flow of the different emergency sectors and help coordination in field operations. The online application; which represents only a fraction of an actual system for the Danish emergency rescue services, will be suitable for pc's, laptops and tablets. Any expressed opinions, recommendations and needs of the Danish Emergency Rescue Services in this report are entirely expressed by us and not the Danish Emergency Rescue Services.

## About the audience of this report

We have written this report for readers with knowledge of geographic information systems and some familiarity to the Danish Emergency Rescue Services. The audience is primarily intended to be our supervisor and the examiner.

## Organisation of this report

This report is organized into 7 chapters; starting off with an introduction to our research and the studies of the need for a new geographic communication tool for the combined efforts of the Danish Emergency Rescue Services, which we completed in our second and third semester. We also set up our problem statement and 5 research questions. Then we go on to the management of this project, and how we have used Scrum as our project management tool. Next we identify the use cases and user stories which describe the expected use of our application, which leads to the outlining of the requirements for our application. In the fourth chapter 'Theory and Methods' we focus on the theories behind and the methods we use to answer our 5 research questions listed in the Introduction. The 4th chapter also includes a section on the theories of testing. Chapter five documents the technical implementation of our system and the practical testing. In chapter 6 and 7 we go through further discussions and the conclusion of this project.

## About this projects supplements

In connection with the progress of this project we have created 2 online mock-ups, representing prototypes of an application for tablets and a pc. Our online mock-ups can be found on the internet at these addresses:

http://mtmgroup4.endersen.dk/
http://mobile.endersen.dk/

Aalborg the 9th of January 2014

# Abbreviations

| | |
|---|---|
| API | ‘Application Programming Interface’ |
| ArcSDE | ‘Arc Spatial Database Engine’ |
| BTS | 'Base Transceiver Station' |
| CERN | In French 'Conseil Européen pour la Recherche Nucléaire'<br>In English 'The European Organization for Nuclear Research' |
| CM | 'CentiMetre' |
| CPR | In Danish ‘Det Centrale PersonRegister’<br>In English ‘The Civil Registration’ |
| CSS | 'Cascading Style Sheet' |
| CSV | ‘Comma-Separated Values’ |
| DB2 | ‘DataBase 2’ |
| DERS | 'Danish Emergency Rescue Services' |
| DEMA | 'Danish Emergency Management Agency' |
| DMI | 'Danish Meteorological Institute' |
| DNS | 'Domain Name System' |
| ECMA | ‘European Computer Manufacturers Association’ |
| ESA | ‘European Space Agency’ |
| ESRI | ‘Environmental Systems Research Institute’ |
| E.g. | 'Exempli Gratia' (For the sake of example) |
| EPSG | ‘European Petroleum Survey Group’ |
| EU | 'European Union' |
| FF | 'FireFox' |
| FTP | 'File Transfer Protocol' |
| GB | 'Giga Byte' |
| GeoRSS | ‘Geo Rich Site Summary’ |
| GERT | ‘Geographic Emergency Rescue Tool’ |
| GEUS | In Danish **‘**De Nationale Geologiske Undersøgelser for Danmark og Grønland’<br>In English ’Geological Survey of Denmark and Greenland’ |
| GID | 'Geographical IDentifier' |
| GIF | ‘Graphics Interchange Format’ |
| GIS | 'Geographic Information System' |
| GLONASS | 'GLObal NAvigation Satellite System' |
| GML | 'Geography Markup Language' |
| GNSS | 'Global Navigation Satellite System' |
| GPS | 'Global Positioning System' |
| GST | in Danish: 'Geodata STyrelsen'<br>In English: 'The Danish Geodata Agency' |
| GUI | 'Graphical User Interface' |
| HCI | 'Human-Computer Interaction' |
| HOB | In Danish: 'HelhedsOrienteret beredskabsplanlægning'<br>In English: 'Prepared Planning and Crisis Management' |
| HTML | 'HyperText Markup Language' |

| | |
|---|---|
| HTTP | 'HyperText Transfer Protocol' |
| JDBC | 'Java Database Connectivity' |
| JPEG | 'Joint Photographic Expert Group' |
| JS | 'JavaScript' |
| JSON | 'JavaScript Object Notation' |
| KST | In Danish 'Kommandostation'<br>In English 'Command station' |
| KML | 'Keyhole Markup Language' |
| ID | 'Identity' |
| IE | 'Internet Explorer' |
| I.E. | 'Id Est' (That is) |
| IIF | 'Inline IF' |
| INSPIRE | 'INfrastructure for SPatial Information in the European Community' |
| IP | 'Internet Protocol' |
| ISO | 'International Organization for Standards' |
| ISP | 'Internet Service Provider' |
| JS | 'JavaScript' |
| LAN | 'Local Area Network' |
| NAVSTAR | 'NAVigation Satellite Timing And Ranging' |
| NOST | In Danish: 'den Nationale Operative STab'<br>In English: 'the National Operative STaff' |
| MAC | 'Media ACcess address' |
| MBBL | In Danish 'Ministeriet for By, Bolig og Landdistrikter'<br>In English 'The Ministry of Housing, Urban and Rural Affairs' |
| MS | 'MicroSoft' |
| OGC | 'Open Geospatial Consortium' |
| OO | 'Object Oriented' |
| PC | 'Personal Computer' |
| PDF | 'Portable Document Format' |
| PL | 'Procedural Language' |
| PNG | 'Portable Network Graphics' |
| POP3 | 'Post Office Protocol version 3' |
| QGIS | 'Quantum GIS' |
| RAM | 'Random Access Memory' |
| REST | 'REpresentational State Transfer' |
| SDK | 'Software Development Kit' |
| SDL | Spatial Data Layers |
| SMS | 'Short Message Service' |
| SPIT | 'Shapefile to PostGIS Import Tool' |
| SRID | 'Spatial Reference System Identifier' |
| SQL | 'Structured Query Language' |
| SVG | 'Scalable Vector Graphics' |
| URI | 'Uniform Resource Identifier' |
| URL | 'Uniform Resource Locator' |

| | |
|---|---|
| US | 'United States' |
| W3C | 'World Wide Web Consortium' |
| WAN | 'Wide Area Network' |
| WCS | 'Web Coverage Service' |
| WFS | 'Web Feature Service' |
| WFS-T | 'Web Feature Service - Transaction' |
| WHATWG | 'Web Hypertext Application Technology Working Group' |
| WiFi | 'Wireless Fidelity' |
| WKB | 'Well-Known Binary' |
| WKT | 'Well-Known Text' |
| WMS | 'Web Map Services' |
| XHTML | 'Extensible HyperText Markup Language' |
| XML | 'Extensible Markup Language' |

# Acknowledgements

We would like to express our appreciation to the 4 people, who have helped us through these last 4 months. In particular we would like to show our appreciation to Henning Sten Hansen for his patience, reviews and guidance throughout the creation of this report and our application. He has guided us from the beginning; asked us to be very specific in our objectives and narrow down our workload to only contain the essence of our goals. We have been very happy and content with all the feedback; which we have received and we have appreciated his advices and counsel.

Henning Sten Hansen, Aalborg University

Morten Fuglsang, Aalborg University
Jan Staunstrup, Aalborg University

Hasse Hauch, Municipality of Frederiksberg

# Introduction

We have observed a need for a common tool within the Danish Emergency Rescue Services, as the individual emergency sectors today use various tools and data. The public emergency services in Denmark, regulated by local authorities, are legally bound to put aside resources for what is called Risk-Based analysis. There are 2 methods of assessing this with the same goal of preparing for unforeseen emergencies:

- The first is a process model that identifies the local picture of risk, analysis assessment, and goes on to analysis and allocating resources to a level of service, which can eliminate the potential local risks.
- The second is 'Prepared Planning and Crisis Management' (in Danish and in this project shortened by HOB) which gives guidelines but is not legally binding for the emergency operators. HOB is divided into 7 main areas: Management, Planning basis, Prevention, Training, Exercises, Assessments and Contingency Plans.

The overall objective of both methods is to expect extraordinary events and to be as prepared as possible for when they arise. Many of the details included in both methods are geographically related. It seems only natural that this information could usefully be incorporated into a geographical information tool, and that the tool can be used to support the preparedness efforts of the emergency response teams. Today geographic information is used to varying degrees in the local emergency responses, from map printouts to local web solutions for emergency situations. However, there is no common standard or legislation to ensure an integrated tool that can be used for overall emergency management in the operational field. Problems can arise in the coordination of different emergency service responses when tackling an extreme weather situation, for example in the communication across municipality borders.

This report is a continuation of our project from the 3rd semester in the spring of 2013 *'Geografisk information som beredskabskommunikationsværktøj ved klimaskabteoversvømmelsesscenarier'* (Cosic, Endersen, Foss & Palmqvist, 2013). This project was based on questionnaires, workshop information and experiences, and also system requirements that users have expressed a need for in connection with extreme weather events leading to floods. Our prior projects have showed that there was and are a need for a comprehensive solution.

The requirements which we have identified for an emergency response GIS tool is the following;

- Reliability
- Intuitiveness
- Prompt response time
- Allowing access to the desired information at the right time.

The Storm Surge over Copenhagen in the summer of 2011 led to a crash of the Internet over large parts of the Copenhagen area.

We have asked ourselves this question:

*How can we develop a spatial information tool that can be used in the field and also keep the flow of information between geographically separate emergency service teams flowing?*

This tool is to be used for individual rescue operations, and also to provide an overview that will aid decision making between the individual emergency sectors. Our tool will only resemble a minor part of the comprehensive system which is needed by the Danish Emergency Rescue Services (DERS). The DERS have many needs, demands and ideas as to what should be included in a software like this. But we have decided to follow our own path; to focus on the needs and regulations which match our tool and our ideas. In the development of our tool we have tested several different theories and methods on our path to creating our tool as an online application named GERT. GERT is an abbreviation of 'Geographic Emergency Rescue Tool'; this will be elaborated on in the following chapters. Some of the theories and methods, which we have tried is using Open Source data and applications like Leaflet and creating an application from scratch. What make our system unique are several features:

- This is a Danish system; developed in Denmark and include data which is stored in Denmark; any improvements can be performed in Denmark.
- We use data from data sets which covers all of Denmark; and which follow the Infrastructure for Spatial Information in the European Community (INSPIRE) and ISO standards (International Organization for Standards), collected from Kortforsyningen.
- When our application is fully functional it is possible to share and receive the common image of the situation, with all the actors at site of an incident of flooding.

Attempts have previously been made to introduce a Geographic Information System into the DERS; this system is called 'GeoConference'. Only the Danish Police department uses GeoConference to this day, this is mainly due to the fact that the producer of GeoConference PCI Geomatics has chosen to stop all further development of the software GeoConference.

Today, it is the Danish Geodata Agency (GST), which is a part of the Danish Ministry of the Environment, who is responsible of choosing and incorporating a new Geographic Information Communications System for the joined DERS.

## 1.1 Summaries of our earlier MTM project reports

Our first MTM project report, from the autumn of 2012 *'Kommunikation med Geografisk Information i det danske beredskab'* (Cosic, Endersen, Foss & Palmqvist, 2012), was considered a pre-analysis, building up to our second MTM project report *'Geografisk information som beredskabs kommunikationsværktøj ved klimaskabte oversvømmelsesscenarier'* (Cosic, Endersen, Foss & Palmqvist, 2013), which has been our stepping stone to this third and final MTM project report.

During our first MTM project we sat out to discover who the potential users were, and which data and technical solutions were needed in order to handle climatic-made flood scenarios. We found that there is a well-defined three level structure for the DERS: Municipal-, Regional- and State Rescue Services. Each level of Rescue Services is encouraged to use the HOB guidelines.

The Municipality Rescue Services are bound by law (BEK. no. 765: 'Risikobaseret kommunalt redningsberedskab' (Gade, 2005) to identify and assess local risks like flooding etc. and plan for the required personnel and material needed to solve the assignment. The Danish emergency system acts after the principles of subsidiarity, which means the local municipalities have the primary responsibility to act. If further resources are needed the next level of emergency may be required. In case of extreme crisis events the National Operative STaff (NOST) is set in action to handle the central coordination between different parts of the emergency sectors.

After the terror attacks in New York 9/11-2001, the DERS has trained every other year with the purpose of reviewing the national emergency system on different crisis scenarios. GeoConference have been used and evaluated in training. Evaluations have shown a lack of focus between emergency rescue sectors regarding coordination.

There are many different branches in the DERS. The lack of legal direction regarding emergency plans, makes it is more difficult to implement a geographic tool to coordinate working partnerships between the various sectors and municipality boundaries.

During our second MTM project we set out to discover different available geographical tools used in other countries. We also tried to determine whether the climate conditions, when it comes to future flooding, are relevant for the DERS to prepare for.

Together with The Danish Geodata Agency (GST), we set out to question a representative section of the DERS to identify their general wishes for a geographical tool and which information layers that is essential for handling incidents of extreme rain and seawater on land. The data layers (information) and tools to be available in case of flooding. The results were non-conclusive regarding DERS's needs when it comes to information and communication. Our survey has shown general indications of the emergency commanders' desire for comprehensive amounts of data. As they say themselves: 'We want to know everything there is to know about everything and everybody in the area of interest, 100% accurately and up-to date.' This is a lot to compress into one system, therefore our aim will be to identify a minimal and yet usable set up to satisfy some of the DERS personnel's needs.

In order to identify a suitable application and an acceptable level of information, our team believes that a further detailed analysis will be required. In making our prototype, we choose to follow the DERS guidelines: 'Retningslinjer for indsatsledelse' (Beredskabsstyrelsen, 2010) to insure the relevance of our application for the users in field.

Our research has shown that there is not a dramatic difference between the basic information layers in the handling of extreme rain and seawater on land because the effect is the same: flooding. Based on the conclusions of the two last projects, the group will incorporate the knowledge gained into this third and last MTM project report. In this report we will focus on 2 user groups, one group of users in the field and another group of users in the NOST (command centre). We will use our findings from the first and second MTM project to determine the system capabilities, look, feel and information supply.

Within the different emergency areas there are a variety of systems in use from the very basic paper maps to sophisticated computer systems. This creates a problem when you have major incidents, an emergency that needs to be handled across municipal boundaries can be handled

in different ways in the different municipalities. There is no central system. An ideal prototype has to fulfil all the different requirements of the different parties involved; municipalities, regions and state.

## 1.2 Objectives - Problem statement and research questions

*The aim of this project is to develop a prototype software system, which will support the communication between emergency response teams by use of geographic information in an emergency situation, connected with catastrophes of flooding and saving lives.*

Furthermore we aim to improve communications by eliminating the administrative border obstacles between municipalities and by sharing a digital map instead of printed maps.

We have approached this problem statement by analysing the following questions**:**

- How can we establish and host a database to support updating?
- How can we design an appropriate user interface for Emergency Response work?
- How can we develop a prototype, a mobile application based on Web standards and open source products, meeting the needs of the emergency rescue teams and NOST?
- How can the developed prototype tool be used as a human rescue and warning system?
- How can we apply Scrum to our system development in a geographically spread environment?

To identify our limitations from the beginning we have had to determine 'what is in our Scope' and 'what is out of scope'. Due to the time restrictions set for completing this project, we have found it necessary to focus on a small software system; resembling only a fraction of the large and comprehensive system, which is in reality needed by the Danish Emergency Rescue Services (DERS). We are well aware that a complete software system for the DERS would include many extra functions and finesses, and also a lot more time for development and more manpower with the needed qualifications.

## 1.3 Target groups and potential users of our application

In real life a communications tool for the DERS, is to be outlined and specified by the Danish Emergency Management Agency (DEMA, a Part of the Danish Ministry of Defence) in cooperation with the Danish Geodata Agency (GST, a part of the Danish Ministry of the environment) in order to achieve the maximum benefits of a GIS software as a communication system. DEMA has the necessary experience and knowledge of emergency situations to interpret and evaluate extreme situations, and they have extended experience with field operations. GST has all the data and GIS knowledge needed to build a system for the emergency response teams.

Our main system is addressed to the personnel in the field, as they are dependent on communication in order to fulfil their job and save lives. As we expect the users to be emergency rescue experts and not GIS experts, our system must provide an intuitive user experience while providing a maximum output to support the user's needs. Cai, Yu & Chen provides the evidence that human-GIS interactions can be possible without requiring algorithmic knowledge about GIS functions (Cai, Yu & Chen, 2013).

What is to be communicated?

Officers acting in the field during critical situations of e.g. storm surges need to have correct and to-the-point information concerning the situation at hand. As it is said 'An image speaks louder than words', a map supplemented with visual information in the form of e.g. polygons representing flooded areas and symbols representing roadblocks, will be quick and easy for the users to update and even quicker to interpret.

In a geographic information system anything with a geographic reference or an address can be communicated. To a field officer some of the most important things to be communicated during a flooding is the following: outline of the focal area, possible development of the flooded area, road network and evacuation routes, best locations of roadblocks, evacuation centres, which areas of the risk zone is to be evacuated in which order, how many households and how large a population lives in the area, etc.

## 1.4 Chapter flow of this report

In short in this report we start off by introducing our project, the tools and the methods which we have used to gain a good base for the project (chapter 1-3). The order of the chapters 4 and 5 are similar, section 1-4/5 in each chapter matches our research questions from section 1.2. In chapter 4 section 4.1-4.4, we identify the theories and methods on which to base our answers. Section 4.5 sets the basis for testing our application and 4.6 is a summary of chapter 4. Chapter 5, section 1-5, describe our work in the attempt to answer the research questions. In addition to answering the research questions, section 5.6 describes the testing of our application and 5.7 is a summary of chapter 5. Chapter 6 and 7 is our discussion and conclusion chapters.

Chapter 1: Introduction

Introducing our project; by summing up our prior conclusions, our problem statement and introducing our users.

Chapter 2: Scrum as our project management tool

Theory of Scrum and our use Scrum

Chapter 3: Use case and requirements

By telling the story of a user story, we identify a use case of our application to be, which brings us to the writing of our requirements outline.

Chapter 4: Theories and methods

In chapter 4 we look into the various theories and methods, which we find to be appropriate towards finding the answers to our research questions.

Section 4.1 here we ask ourselves the question '*How can we establish and host a database to support updating?*' looking into the system requirements, Security and maintenance.
Section 4.2 addresses the question '*How can we design an appropriate user interface for emergency response work?*' including Human-Computer Interaction (HCI) and the prospects of our user interfaces.
Section 4.3 focus on the question '*How can we develop a prototype, a mobile application based on Web standards and open source products, meeting some of the needs of the emergency*

*rescue teams and NOST?*' to enlighten our project statement from chapter 1. These are the points we will follow: System Architecture, Input Data, Metadata and Programming.
Section 4.4 looks into the question '*How can the developed prototype tool be used as a human rescue and warning system?*' we go into geolocation, Global Positioning Systems (GPS), IP addresses (Internet Protocol addresses), Cell ID (Identity), WiFi (Wireless Fidelity) and Bluetooth Mac (Media Access) Addresses.
Section 4.5; insight into '*Theory and methods of software testing.*'
Section 4.6 sums up on the contents of chapter 4.

### Chapter 5: Technical Implementation

Chapter 5 concerns our work and results, based on the theories and methods from chapter 4, to satisfy our problem statement and research questions.

Section 5.1 addresses updating and how to share a database.
Section 5.2 displays our progress towards designing appropriate user interfaces for emergency response teams.
Section 5.3 describes data and databases, our experiences of publishing data, and the equipment to be used.
Section 5.4 is a display of the techniques we have used and our programming.
Section 5.5 displays our solution to a rescue and warning System.
Section 5.6 documents our performed testing.
Section 5.7 is a summary of our experiences documented in chapter 5.

### Chapter 6: Discussion

Discussions of our theories, methods, findings, reflections, and perspectives.

### Chapter 7: Conclusion

Conclusion of our findings and experiences

As part of this project we aim to use two independent servers in two locations with the same implemented solutions. By doing this we will try to secure the project against possible failure caused by crash or other kinds of malfunction. Even though synchronizing the system will add to the workload, it is our belief that this time is well spent.

# 2 Scrum as our project management tool

We have already introduced Scrum as our project management tool in our previous projects. Therefore, in this chapter, we will give a short introduction on Scrum as a method for software development and to how we have used Scrum during the development of our project this semester.

Scrum is an agile software development method, which is based on iterative and incremental software development. At the start of this semester we used an excel sheet to manage our work, which made it possible to register our product team, user stories, time estimates and the remaining time. We knew it could help us manage our work, but we lacked visualization while using excel, therefore we sought for another appropriate solution. After getting in contact with Axosoft[1] we got an offer to be introduced to their product for free, this software is designed for agile software development. The application is called OnTime and it is strong and intuitive software.

For this project we introduce the Agile Manifesto for our software development which contains four elements which we explain briefly here:

> *'Individuals and interactions over processes and tools'*
> *'Working software over comprehensive documentation'*
> *'Customer collaboration over contract negotiation'*
> *'Responding to change over following a plan'*
>
> The Agile Manifesto[2]

- ***Individuals and interactions over processes and tools:*** Scrum is an agile method (process). Any tool to be used has to be simple and easy for the entire group to use, for interacting with each other and solving problems as a group, and not slowing down the team with too many tools and processes.
- ***Working software over comprehensive documentation:*** To work with the software is paramount, but some documentation is also of importance, to keep the software in order and prevent the system ending up as a big giant of mass of nothing.
- ***Customer collaboration over contract negotiation:*** Despite the fact that not everybody sells their product, the concept still stands. Customers are the people who use the product. Users, who have experienced the product in question, often have comments related to their experiences; which will be of use to the developers. Customer feedback is a very important to method to receive feedback and to take in customer response with regards to improvement.

[1] www.axosoft.com
[2] http://agilemanifesto.org/

- ***Responding on change over following a plan*:** The purpose of planning is to make sure that we know and follow the same direction throughout the project. But above all still be flexible to changes and refitting of the common plan.

The first three points are the most important and help us respond to changes by following a plan, which is part of the 4th point. Implementing these 4 points of the Agile Manifesto is described in section 2.1.

## 2.1 Implementing the Agile manifesto for software development in our project

In this section we introduce the Agile Manifesto for software development in our project, and identify our gains by the implementation.

- ***Individuals and interactions over processes and tools:*** One of the conditions for using Scrum is a daily group meeting, as a group we live far away from each other and we all have day jobs, so we don't have opportunity to fulfil this demand. We have chosen to use Skype as our communication tool between both internally in the group and externally to communicate with our supervisors. Instead of daily meetings, we met at minimum once a week on Skype, where we talked, shared documents and links. We found this is a good method for us, and that Skype fulfils this requirement.
- ***Working software over comprehensive documentation:*** It is very important to preserve all documentation during the implementation of software. We have had to be careful to keep the documentation short and precise, to prevent slowing down the combined work of the group. It should always be possible to go back and recall a document if needed. We have used GOOGLE DRIVE as a common archive to keep track of all our documents. It is online and available to all of us at any time, and from any location by mobile phone and pc (Personal Computer). The only requirement is an internet access. All of our Skype meetings with our supervisors, have, with their permission, been recorded and saved in separate files along with all our documents, which makes it easy for us to find them again.
- ***Customer collaboration over contract negotiation:*** Our customers are the users who we have designed our product for (in this case the emergency commanders). The Scrum features are written from the perspective of the end users, which is why we have built our product in accordance with the user stories. During our third semester we collected many user stories by interviews and participating in workshops. These user stories have been fundamental for building our application.
- ***Responding to change over following a plan:*** The purpose of planning is to ensure that we all recognize the direction in which we are heading. We have proceeded with this project with good collaboration following the scheduled plan, while still being flexible enough to follow the project when we discovered new and better directions.

The main gains which we gotten by following the manifesto are remembering to document everything in the common archive, and staying flexible to follow the project when the path twists and turns.

## 2.2 Our use of Scrum methods

We have implemented Scrum in our project and found that Scrum has been a useful tool for our project. We have used Scrum to fit our conditions and requirements. In this section we identify the Scrum roles in our project and describe the implementation of Scrum tools in our project.

There are three roles of Scrum; the Product owner (who represents the customers), the Scrum master (paving the path and guiding the members towards the goal), and the Product team (members of the development team).

Scrum roles in our project:

- **Product owner:** As this project have not been developed for a real commercial purpose nor ordered by a real person or company, we have taken the Product owner position on ourselves. We have conducted interviews and participated in workshops with the intended users and created a product backlog from user stories.
- **Scrum master:** Our supervisor (Professor H. S. Hansen) has filled that role by guiding us, his useful and professional advices has kept us in the right direction toward the main goal. We have had continuous contact with him during the implementation of our project.
- **Product team:** The product team are those who are technically involved in producing the software. As a MTM group; we have worked together to develop an application, making it possible for us to reach our goal.

The Scrum process consists of three tools; the Product backlog, the Sprint backlog and a Burndown chart.

Scrum Tools in our project

- **The Product backlog:** Lists all features, functions, requirements, enhancements, and fixes that constitute the changes to be made to the product in future releases. Product Backlog items have the attributes of (Schwaber, Sutherland 2011):
    - Description
    - Order
    - Estimate
- **Sprint backlog:** Represents time management for software development. The time periods is normally set at 1 to 4 weeks. It typically starts by identifying the user stories, which are to be a part of the release backlog. Using sprints helps tackle manageable parts of the project and get the work done.
- **Burndown chart:** Monitors the progress of sprints and helps ensuring the product progressing running smoothly. The Burndown chart provides a day by day measure of the amount of work remaining in a given sprint, and helps visualize the progress and ensuring that the team is on the right track.

To start this project we created our Product backlog upon a collection of wishes expressed by the potential users of our product; *the Customers,* from our last project. This Product backlog was gained through interviews and workshops with users which we had saved in our Google drive archive. We set out to complete our project in four sprints each of six week periods. Figure 2-1 illustrates the Scrum process; the path from a user story in the product backlog, into several sprint backlogs, and from there into the sprints of development until it finally ends as a working implemented software.

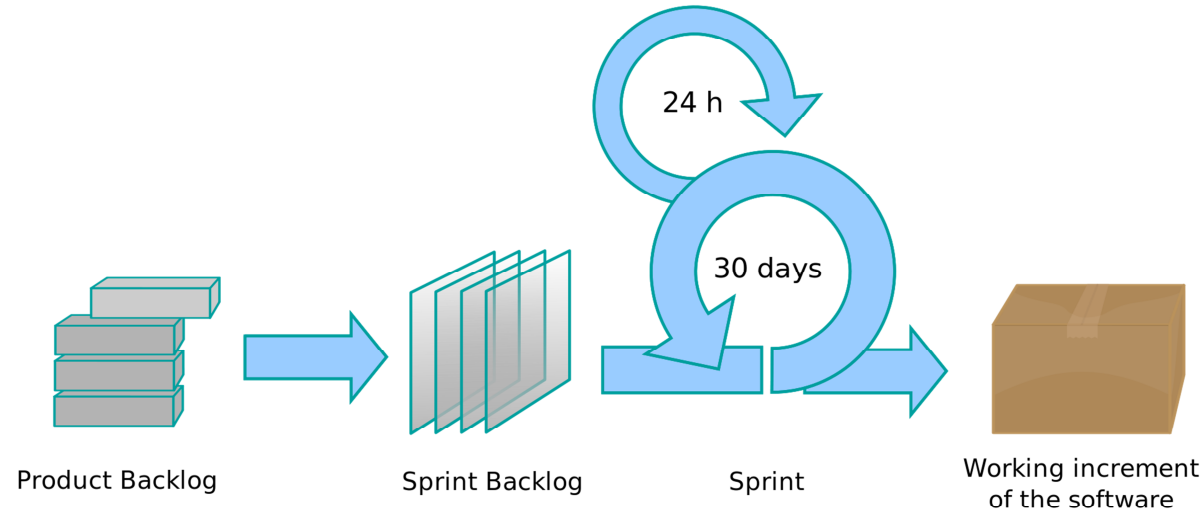



**Figure 2-1:Scrum process, shows how a user story from Product backlog is broken down into small parts called Sprint backlog, afterwards it is ready to be worked out in a sprint process for ultimate shipping (Wikimedia)[3]**

Scrum is generally used to track the number of days used to complete a task. But as we all have daytime jobs, we decided not the register in days with a fixed number of working hours per day. Instead we register our work in tasks of similar duration.

The preliminary sprints of our project:

- **1st Sprint:** Identifying the project objectives and literature study, installation and adaptation of the system, web server, GeoServer and database. Identifying a system setup to meet our requirements and capabilities.
- **2nd Sprint:** Specifying the objective of this report and building the foundation for our application and this report.
- **3rd Sprint:** Collection of data and implementing it in our system, application development, creating a web setup with user interface and writing the report.
- **4th Sprint:** Implementing and testing the application. Finish writing the report.

Implementing Scrum has been a useful tool for this project.

## 2.3 Our experience of using methods from Scrum

We have used OnTime as a source of inspiration to implement Scrum methods in our project. By introducing methods from OnTime, we have gained assistance in keeping an overview of our project developments.

OnTime is divided in three main sections:

1. Organisation visualizes the four major structures in OnTime. These structures are the Project tree, Releases, the Team members and Customers.

[3] http://upload.wikimedia.org/wikipedia/commons/thumb/5/58/Scrum_process.svg/2000px-Scrum_process.svg.png

2. Managing the Product backlog by adding, removing and editing user stories, setting estimations and calculating the remaining time, prioritizing, and reading and writing descriptions.
   There is also a 'Defect Backlog' function, when a product is rejected in the testing stage; it comes back under the 'Defect Backlog' for further work. A Burndown chart is made automatically in OnTime, it is updated when something changes in our work and we can print it out as a PDF (Portable Document Format) file.
3. Detail on specific items selected in the main workspace: adding to and editing in the work log, attaching documents to specific items, managing sub items, e-mails, history etc.

In this chapter we have through our use of Scrum for our project. How we have modified Scrum to best meet our working conditions, meetings at least once a week, documenting our progress, guidance by our Scrum master, and used inspiration from the OnTime methods to keep track of the project development.

For every meeting we all report on our personal progress, so that all are updated on the current state. Here is a short list of the fixed topics which are always part of our meetings:

- What have been done since we last meet, what are we working on, which issues have been faced – who can help
- Meeting deadlines and milestones, Burndown chart
- Product backlog, Sprint backlog

We started by writing down everything we could think of; in regards to our project scope, and created a list of all this things needed be done to commence the project. This formed the basis for our Product backlog and that was the start for our first sprint. As this has been an agile project, we did not set out with a fixed set of requirements. We have allowed the requirements to change along the way, and view the product backlog as a living artefact.

As we, as a study group, live in four different geographic locations, we lack the opportunity to meet daily. The daily meetings are one of the demands set by the Scrum method. Instead we have met at least once a week on Skype. In regards to the Product owner and the Product master, we view these figures as fictive as this is a study and our product is only regarded as a prototype. We have documented our processes through this project.

We have used Scrum in the best possible way, which has been appropriate for our situation. As we have used the agile method on all the areas of our project, adjustments have been easily introduced and incorporated at the appropriate stages of the project.

# 3 Use cases and requirements

We have used Scrum, an agile software management tool to help keep track of this project. In addition to this; we have not had a complete list of requirements for our application to begin with. Instead our agile method has led us to our requirements by assessing use cases and the stories behind. This method has let us start on the software development at an earlier stage of the project, than if we had started out writing a comprehensive documentation of our requirements. In short, our requirements would not have had room to change and grow during the development of our project. This method ensured that any changes to our software solution could be incorporated in an appropriate manner. Our use cases documents the use in short direct stages.

In the Scrum environment, it is the Product owner who gets together with a customer representative for creating user stories. The customer has the responsibility for formulating user stories. The Product owner may use a series of questions to get the customer going, such as asking about the desirability of some particular functionality. In our case we have set out being the Product owners and our customers are fictional figures. We have created our own user stories in accordance with our experiences from our two previous projects mentioned in the introduction.

> 'A usage *narrative* is a situated example of the use case in operation - a single, highly specific example of an actor using the system.'
>
> '... invent a fictional but specific actor and briefly capture the mental state of that person - why he wants what he wants or what conditions drive him to act as he does.'
>
> 'Brevity is important so the reader can get the story at a glance. Details and motives, or emotional content, are important so that every reader... can see how the system should be optimized to add value to the user.'
>
> (Cockburn, 2007)

By studying usage narratives; also called user stories, we aim to get an in-depth feeling of our users and their stories. The user story is then used to build our software requirements upon. The desired outcome of this chapter has been to design the requirements of our system; the processes to lead us there are our user story and use cases.

## 3.1 User stories

After reading about Alistair Cockburn's (Cockburn, 2007) methods of how to write user stories, we have created our own recipe and a useful user story. The basic ingredients of our recipe can be found in the box below.

Our user story concerns an emergency commander from DERS, whose job it is to assess the primary situation in the field in connection with emergency and rescue operations, and report back to the NOST with updates. The structure of our user stories consist of: *Actor*(s), *Action*(s) and *Achievement*, and *Motive*(s) and *Emotion*(s).

'Because of <**Motive**> I as an <**Actor**> want to <**Action**> so that <**Achievement**>, by doing this I feel <**Emotion**>.'

***Actor:*** The owner of the user story (often a user). It is very easy to end up using the name 'user' for the actor, but it is recommended to be more specific. By using specific actors it is easier to understand and set the user story in context with the system.
***Action:*** What does the actor want to do? It is also possible to differ between mandatory actions and optional actions. This can be done by using the *want* or *must* keywords before the action.
***Achievement:*** What does the actor want to achieve by performing this action? This is the result from executing the action seen from the actor's point of view.
***Motive:*** Why the actor wants what he/she wants? Which conditions drive the actor to act in this way?
***Emotion:*** Will this make the actor feel good? Does the actor have a good feeling about this action?

Here are three short user stories which are to be used to base our use case on:

1. 'Because of <**flooding**> I as an <**emergency commander**>; want to <**define the focal area**> so that <**the focal area can be visualized**>, by doing this; I feel <**content that all the emergency response teams see the same flood outlining as I**>.'

2. 'Because of the <**risk of personal injury**> I; the <**emergency commander**> want to <**place roadblocks**> so that <**no civilians enters the focal area**>, by doing this I feel <**content that no one will get hurt**>.'

3. 'Because of <**flooding**> I; the <**emergency commander**> want to <**identify civilians in need of evacuation**>, so that <**I can inform and evacuate the civilians**>, by doing this I feel <**content that all civilians will be safe**>.'

An example of a more comprehensive user story could be a follows:

'The emergency commander drives to a focal area, for example flooding that has been reported to the emergency call centre. On arrival the emergency commander assesses the affected area and uses the geographic emergency system to get an overview of the neighbourhood and the adjacent areas. By analysing the flooding and evaluating any further risks, the emergency commander registers the focal area in the geographic emergency software thus sharing his/hers knowledge with the cooperating emergency response teams. Then the emergency commander sets up roadblocks to ensure that no civilians will wander into the affected area. These roadblocks are inserted into the geographic emergency software, to be shared with the colleagues. Next evacuation of the civilians within the affected area must be started. The emergency commander starts by assessing the area from where all the civilians must be evacuated from. Then the emergency commander defines the evacuation area in the geographic emergency system. Using the select tool, he/she can identify all residents who need to be evacuated. Having evacuated all civilians and communicated all the necessary information on the affected area the emergency commander feels the situation is under control.'

In these user stories we have attempted to follow the guidelines of the Danish Emergency Management Agency (DEMA). An abbreviated and translated list of the DEMA guidelines and

action sequences can be found in Appendix 1. In the next section we build use cases on our user story to create the basis of our software Requirements.

## 3.2 Use cases

We use 'Use cases' to identify the requirements of our application, and describe the requirements to be met by our application prototype. 'Use cases' can help focus the discussions about upcoming software system requirements (Cockburn, 2007). Writing our use cases is a way for us to create our vision of the system (Cockburn, 2007). According to Alistair Cockburn (Cockburn, 2007) two things must be kept in mind when writing 'use cases' as requirements:

> - *They really are requirements*. You shouldn't have to convert them into some other form of behavioural requirements. Properly written, they accurately detail what the system must do.
> - *They are not all of the requirements*. They don't detail external interfaces, data formats, business rules, and complex formulae. They constitute only a fraction (perhaps a third) of all the requirements you need to collect - a very important fraction but a fraction nonetheless.
>
> (Cockburn, 2007)

'Use cases' are small structured descriptions, describing how a user is supposed to act in a given situation to achieve a certain goal. It is important to keep the descriptions as simple and understandable as possible. Making 'use cases' is an iterative process where it is possible to work on and refine the 'use cases' at later stages.

The use cases; which we focus on in this project are concerned with the behavioural requirements of our software. The stakeholders are the people who we expect to be the users of a completed application, the owners (in this case our project group), government regulatory agencies, and other computer programs (Cockburn, 2007). The primary users of our software are expected to be field officers, emergency commanders and users at the NOST. Our primary users/actors are officers from the DERS. We are familiar with some of their requirements to communicating geographic information during emergency situations from our prior projects and the HOB documentation.

> 'A use case captures a contract between the stakeholders of a system about its behavior. The use case describes the system's behaviour under various conditions as the system responds to a request from one of the stakeholders, called the primary actor. The primary actor initiates an interaction with the system to accomplish some goal. The system responds, protecting the interests of all the stakeholders. Different sequences of behavior, or scenarios, can unfold, depending on the particular requests made and the conditions surrounding the requests. The use case gathers those different scenarios together.'
>
> (Cockburn, 2007)

For our system, we have selected three different functions, the result of which are to be communicated out to the other users including the NOST, and logged for later evaluation. A statement can be visualized by use case diagrams, showing for example how the emergency commander interacts with the system. The diagram provides a simplified and graphical representation of what the system (software solution) must be able to do.

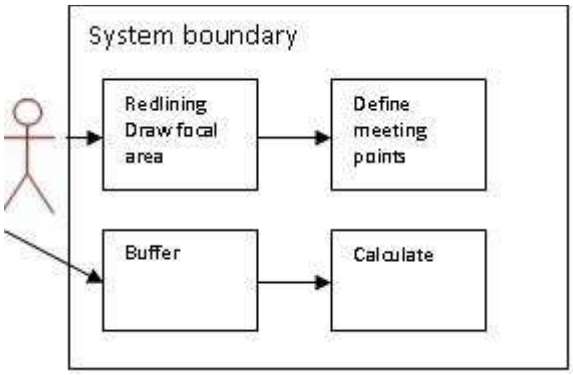



Figure 3-1: Use case example, resembling how one action is followed by another as part of a sequence (MTM group 4)

'Use cases' could be a sequence of events. In the above 'use case' example shown in Figure 3-1, one action is followed by another action; 'Define meeting points'. For our 'use case' we introduce a use case Template based on the use case descriptions from 'Writing Effective Use Cases' by Alistair Cockburn.

***Describe the case in one line, a contract for the behaviour of the system***

**Primary actor:** *Stakeholder – who has the goal?*

**Scope:** *What is really the system under discussion? Identifies the system*

**Level:** *How high- or low-level is the goal?*

**Stakeholders and interests:** *Someone or something with a vested interest in the behaviour of the system under discussion*

**Pre-condition:** *What must be true before the use case runs?*

**Minimal Guarantee:** *What must be minimally true after the use case has run?*

**Success Guarantee:** *What must be true after the use case has run?*

**Main Success Scenario:** *What defines our total success of this use case?*

**Extensions:** *What can happen different during the scenario? Numbering in the extensions refer to the step numbers in the main success scenario at which each different situation is detected.*

(Cockburn, 2007)

Our 'use case' has been based on the comprehensive user story, described in section 3.1. This is to identify the expected response of the system to the primary actor's actions. The 'use case' could be split into 3 or more simplified use cases to match the user stories number 1, 2 and 3.

In Appendix 2; short use cases on NOST users and evaluation can be seen.

***Reporting and sharing the status of a flooding and the rescue operations***

**Primary actor:** The emergency commander

**Scope:** Sharing the geographic 'image' of an emergency situation, drawing and sharing the focal area, drawing and sharing the location of road blocks, and finally drawing a polygon to select the number of residents within the geographic area.

**Level:** High level user goals.

**Stakeholders and interests:**
The emergency commander
Emergency personnel
NOST users

**Pre-condition:** Our application must be installed. Online access to databases is also a necessity.

**Minimal Guarantee:** Drawing in the map and sharing the results

**Success Guarantee:** All parties involved in the operation can see the drawn objects in their geographical emergency tool and the changes have been logged with geographic references and a timestamp.

**Main Success Scenario:**

1. Open the geographic emergency tool
2. Using the button ‘Get Location’ identifies the primary actor’s geographic location and zooms in automatically
3. Draw focal area
4. Place a road block
5. Identify the local residents affected by the situation

In connection with points 3-5 the information is shortly shared with the other users.
Extensions can be found in Appendix 3

## 3.3 Requirements outline

As stated earlier in this chapter, use cases can be used to identify our system requirements. Our use case from section 3.1 detail what our system must be capable of, without going into details of external interfaces, data formats, business rules, and complex formulae (Cockburn, 2007).

Our application’s primary user is the DERS Teams and the NOST. Therefore it is our aim to partly match an actual software package for the Danish emergency agencies. We aim for our requirements to match against some of the functions provided by the Danish Geodata Agency (GST).

> '... using use cases to *discover* requirements leads to higher-quality functional requirements. They are better organized and more complete.'
>
> Steve Adolph quoted by Cockburn (Cockburn, 2007)

According to a project proposal; written by the GST on the 23rd of January 2013 (GST, 2013), a large part of coordinated operations, between the emergency rescue services in Denmark, when planning, operating and evaluating, involves sharing the same geographic overview of the situation. Sharing the same geographic source containing current, relevant and credible information, which everyone from the various agencies can see simultaneously, has great value. Some of these are listed below.

- All the involved teams will have the same overview of the affected area
- All the layers implemented will serve the same objects on-screen for all the involved teams
- The objects displayed will have the same styles and features when view by team A and team B

The primary purpose for a Software product for the Danish emergency agencies was originally set by the GST to match the HOB-model created by The DEMA. Some of the emergency guidelines set by DEMA are listed in appendix 1.

## 3.4 Our requirement outline

Organized according to Cockburn (Cockburn, 2007):

1) ***Purpose and scope***
   - ***a)*** The overall scope and goal of this software is to support the communication between emergency response teams by use of geographic information in an emergency situation. With teams using the same software and thereby sharing the same maps, tools, information and knowledge.
   - ***b)*** The stakeholders of this system are the Danish Emergency Rescue Services (DERS), in particular the emergency commanders, who are the field officers in charge and the experts located at the NOST. Secondary all the residents at or near a focal area will be depending on the shared information between the NOST and the emergency commanders.
   - ***c)*** The scope of this software is to be able to share a common area, a focal area, the location of roadblocks and to 'identify residents' by address points within a localized area.

2) ***Terms used / glossary***

| | |
|---|---|
| Focal area | Location of a situation of emergency |
| NOST | National Operative Stab |
| DERS | Danish Emergency Rescue Services |

3) ***The use cases*** *(see the section on use cases on the previous pages)*
   - ***a)*** The primary actor is the emergency commander, and the general goal is to communicate information on a focal area to the NOST and cooperating emergency response teams.

***b)*** Better communication will help the various parties be better prepared and thereby perform better.
**c)** Placing roadblocks to indicate that movements beyond these points are connected to risks.
**d)** Identifying address points to get an indication; as to where civilians live.

***4) The technology used***

***a)*** What technology requirements are there for this system?
**i)** Internet application to be used online and offline.
**ii)** Users in the field use and tablet with a 10 inch screen.
**iii)** NOST will have large screens.
**iv)** Free data and software: OpenGeo Suite, data from the Danish Geodata Agency (GST).
**v)** Background map (easy to interpret).
**vi)** Layer showing flooding areas, risk assessment.
***vii)*** Modification of vector layers.
***1)*** Polygon layer to illustrate focal area.
***a)*** Draw polygon.
***b)*** Modify.
***c)*** Delete.
**2)** Rectangle to identify area.
**a)** Draw rectangle.
**b)** Modify.
**c)** Delete.
**3)** Polylines.
**a)** Draw polyline.
**b)** Modify.
**c)** Delete.
***4)*** Point layer marking roadblocks.
***a)*** Draw new.
***b)*** Move.
***c)*** Delete.
**viii)** Using Address points to identify population.
**1)** Marking the area of interest and identifying the affected address points.
**a)** Select.
**b)** View selection.

***b)*** What systems will this system interface with, with what requirements?
**i)** OpenLayers.
**ii)** GeoServer.
**iii)** PostGIS.
**iv)** Web GIS server at Kortforsyningen.
**v)** Web server.

***5) Other requirements***

***a)*** Development process

- ***i)*** The project participants are Zijad Cosic, Tina Endersen, Bjarke Foss and Azad Palmqvist.
- ***ii)*** The software will be kept simple and intuitive.
- ***iii)*** As we do not have either users or sponsors involved, we are aiming to send feedback on our progress to our project supervisor minimum once a month.
- ***iv)*** Mainly use of Open Source, and we build the rest.
- ***v)*** Testing
- ***vi)*** Online access to databases on server
    - ***1)*** In case of fluctuating internet access and offline working mode must be available.
- ***b)*** Not interfering with HOB guidelines.
- ***c)*** Fast and smooth performance
- ***d)*** Operations, security, documentation
    - ***i)*** Minimum of 2 Danish servers hosting the data, data storage.
    - ***ii)*** Hacker safe *(not implemented)*.
    - **iii)** Documentation of progress during development.
- ***e)*** Use and usability must be intuitive and effortless.
- ***f)*** Maintenance and portability
    - **i)** Maintenance is not relevant as this is a prototype
    - **ii)** Tablet with online access and possibility of offline mode, screen size of minimum 10 inch.
- ***g)*** Unresolved and deferred
    - ***i)*** Scenarios performed by the actor will be shared with the other teams who are assigned to this operation. All scenarios performed on the system will be logged for later evaluation (not implemented).
    - **ii)** The users must be able to share their information and see the shared information of others (not implemented).

### *6) Human backup, legal, political, organizational Issues*

- ***i)*** Human backup to system operation.
    - **1)** Trained technical operators (e.g. NOST users) will be online to aid assistance.
- ***ii)*** Legal and political requirements.
    - **1)** None as this is a prototype.
- ***iii)*** Human consequences of completing this system.
    - **1)** It is a possibility that the users trust more in the information on the screen than on the actual conditions in the affected area.
    - **2)** Overlooking lives at risk, due to incomplete data sets.
    - **3)** A new workflow must be incorporated.
- ***iv)*** Training requirements.
    - **1)** All users must complete a training course and be acquainted with GIS.
- ***v)*** What assumptions, dependencies are there on the human environment?
    - **1)** No assumptions.

The template used for our requirements outline can be viewed in Appendix 4.

## 3.5 Summary of our user story, use cases and requirements

From the user stories we know that our users; the emergency commanders needs a tool which can help them feel content with their decisions and actions; in relation to situations of extreme rain causing flooding and risk of lives. The use cases have identified the users' expectations to the use and behaviour of a system. We have used these use cases to outline the requirements of our software application, and to state that the use cases are, in general, relevant when designing software. We now have a good sense of the data necessary to keep the users well informed, and we have identified the functions which need to be included in our software application. From this point on, we keep our requirements outline as a base to build our software application upon.
In the next chapter we go through theories and methods suitable for reaching our goals for our software application.

# 4 Theories and methods

In this chapter we aim to theoretically cover the remaining 4 research questions from the Introduction in section 1.3. In addition we will also include a section on Software Testing, which we believe will improve the prospects of our application.

- 'How can we establish and host a database to support updating?'
- 'How can we design an appropriate user interface for emergency response work?'
- 'How can we develop a prototype, a mobile application based on Web standards and open source products, meeting the needs of the emergency rescue teams and NOST?'
- 'How can the developed prototype tool be used as a human rescue and warning system?'
- 'Theory and methods of Software testing'

Later in chapter 5 we document our own progress and experiences gained during the evolution of this project.

## 4.1 Ensuring all users direct access to the databases

During rescue operations time is precious, passing time and concurrent delays; can mean the difference between life and death. The system should be fast and intuitive both in communication between involved actors, and the management of data. The communication between the involved actors is mainly dependent on the internet. In remote areas the internet speed can be low, and sometimes internet connections are completely absent. Some of the main data should therefore be preinstalled in the device for two reasons:

1. In the absence of internet connectivity, the ground officers should have access to the most basic data to complete their job.
2. Waiting for transference of large amounts of data by the internet takes a long time and this will decrease the efficiency of the rescue operation.

To get a better insight into the essentials of a real-time system we have looked into the experiences documented by Ko Ko Lwin and Yuji Murajama (Lwin & Murajama, 2011). Lwin and Murajama have developed a system; where students with GPS embedded mobile phones or a GPS and a mobile phone, have collected data in the field. The data collected was then sent as a mail to a POP3 (Post Office Protocol 3) mail server and converted into Microsoft (MS) Access Database format automatically. In the web-GIS system the MS (MicroSoft) Access database is converted into an ESRI (Environmental Systems Research Institute) shapefile. End-users can then update their up-to-date data. The abilities of this system matches to a degree what we are looking for, except that we want our users in the field to be more than data collectors, we want them to also be end-users, preferably without having to update the application manually. The application should communicate with corresponding real-time server during every user connection. The real-time server acts as network operating system for the web applications, attends connections from users, receive its object class and made them a 'live' in the server environment. Intercommunication procedures between online applications, online users and connected devices are settled in the corresponding source code in the real-time-interpreted programming language.

GeoServer comes with a REST API, (REpresentational State Transfer, and Application Programming Interface) and uses operations: GET, POST, PUT and DELETE defined from the HTTP protocol (Lacovella & Youngblood, 2013). The REST configuration API reference for supported formats can be seen at http://docs.GeoServer.org/stable/en/user/rest/api/index.html. It is possible for the client by the HTTP request to receive layers, modify or add features with the GeoServer's REST API, see Figure 4-1.

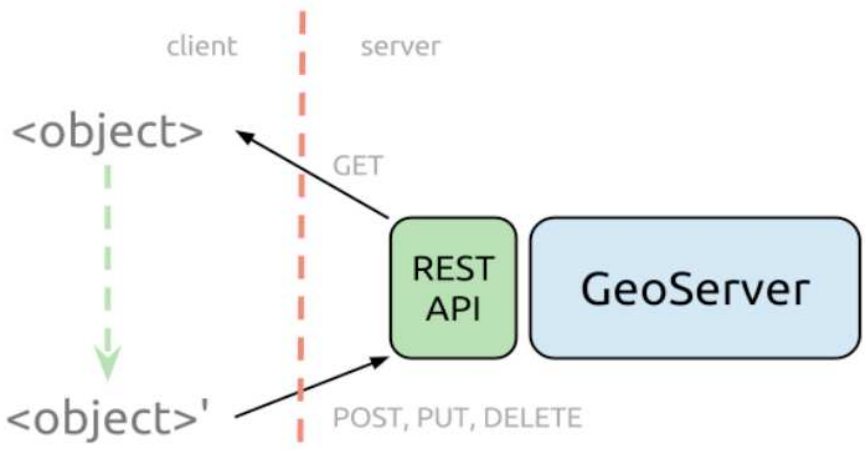



Figure 4-1: Interaction with the REST API (Boundless)[4]

Using the REST API on the client side in combination with WFS-T service, it should be possible to manage data sets in real time. The GeoServer configuration API is placed at http//:<server>/GeoServer/rest in our case at http://localhost:8080/GeoServer/rest. From here it is possible to overview available data on the GeoServer in HTML and viewing the URL for addressing data. We found an example on how to modify WFS-T with REST[5]; the code example can be seen in appendix 5. We haven't used REST in our solution to insure real time data on GeoServer and our application.

## 4.2 Designing user interfaces

According to Brandon Plewe in the book 'GIS online' (Plewe, 1997): '*A well designed interface will be attractive, functional, easy to understand, and easy to use.'* Our approach to a user interface for emergency situations is in-line with Brandon Plewe. The user Interface must be foreseeable, user friendly and self-evident, grant easy access to information, and further information only a few clicks away.

> 'Using a site that doesn't make us think about unimportant things feels effortless'
>
> Steve Krug (Krug, 2000)

An interface design for software like ours consists of two major elements (Plewe, 1997); the user interface and the map which will display our spatial data, including buttons and fields to aid navigation in the map and information retrieval.

[4] http://boundlessgeo.com/2012/10/adding-layers-to-GeoServer-using-the-rest-api/
[5] http://www.ibm.com/developerworks/web/library/os-GeoServer/index.html

Our user interface aims to include elements that perform at minimum three types of functions (Plewe, 1997): navigating or browsing the map, controlling the map's appearance, and performing GIS queries or analyses. By following five simple guidelines listed by Steve Krug (Krug, 2000), we should be able to design an appealing user interface to help emergency rescue personnel to take fast and well-funded decisions. Below are listed 5 simple guidelines to making a design appealing by Steve Krug (Krug, 2000).

1. ***Create a clear visual hierarchy on each page***
    - *First impression*: Map, state of the situation, navigation.
    - *Second impressio*n: Use of layers to form information.
    - *Third impression*: Extracting further information.
    - *Fourth impression*: Communication.
    - *'Creating a visual hierarchy will guide the user through the application and the functions to optimize the user experience and the user results.'*
2. ***Take advantage of conventions***
    - Use of well-known symbols, i.e. + -, etc.
    - *'Use of well-known symbols and icons will improve the user experience, no need to speculate on the functionality of a particular button.'*
3. ***Break pages up into clearly defined areas***
    - Helping the users focus on the important areas of the user interface and identifying the intended use of each area.
    - *'This will help the users to prioritize and identify the usage available through the user interface.'*
4. ***Make it obvious what's clickable***
    - Clear differentials between clickable links and buttons compared to plain text and other objects.
    - *'The user will not be confused as to where the buttons are located.'*
5. ***Minimize noise***
    - Keeping down distractions and unnecessary information.
    - *'Making an effort not to distract the users, aiding them to keep focus on what is important.'*

Counting clicks to get from a starting point to the desired result can be a good way to achieve knowledge on the users' experience of the user interface. But at the same time the amount of thought going into each click has to be considered as well. According to Steve Krug (Krug, 2000) clicks related to mindless choices are preferable to clicks demanding thought. This is related to the amount of uncertainty about whether the user is making the right choices or not.

> 'Experts are rarely insulted by something that is clear enough for beginners. Everybody appreciates clarity'
>
> Steve Krug (Krug, 2000).

By aiming the design towards the primary users and lowering the technical barriers to their standards (Cai, Yu & Chen, 2013), the amount of training needed will be reduced (costs), and the usability will be improved.

#### 4.2.1 Human-Computer interaction

Focusing on the Human-Computer Interaction (HCI), the human user is placed at the centre stage (Worboys, 1995).

> '... HCI... includes an understanding of the psychology of humans when interacting with machines, training, organization and health issues.'
>
> (Worboys, 1995)

For this project it is assumed that the users have sufficient experience with computers to consider computers and tablets as tools and aids in the everyday life. An extensive and comprehensive software system requiring a lot of concentration and clicks can cause the users unnecessary stress. During intense and stressful situations the last thing the users of a communication system needs is extra stress. Work-related stress is considered a dominant health issue. The 'machines' must be the users' friend, the user must have prior experience with the software and preferably training. A geographic communications software program to be shared by many users must be backed-up by the entire organization.

According to Preece et al. quoted by Worboys (Worboys, 1995) there are three components to a successful design:

1. User centred: Users should be involved at all stages of the design process (*letting the users formulate their issues and wishes, give inputs and comments*)
2. Integrated: integrate expertise from several disciplines (*emergency experts, GIS experts, programming experts, etc.*)
3. Iterative: When the design process is iterative, users and the integrating disciplines in the previous step can get involved at any stage as needed. (*analysis, designing, implementation, evaluation, prototype, testing*)

A Graphical User Interface (GUI) with well-known images/icons will assist the users with recognisability and clarity. Therefore when deciding whether to use text, images or an already familiar icon on a button, it is necessary to keep in mind; which information is to be visualized on the particular button. Is it necessary to include a long line of text, a short text, and a photo or is there already a familiar icon known to all who are acquainted with computer icons?

A complete user interface of a geographical Information system to be used as a communication tool for emergency response teams will need a great deal of consideration (Worboys, 1995):

- *Dialog*: Keyboard (for commands, forms and menus), mouse (selections, collections, translations, etc.), Pen (character recognition), Voice (voice recognition) and Screen (touch sensitive screen)
- *Data selection:* by commands or pointing out
- *Data representation*: Areal outline (polygon), Areal fill (area), Line (polyline) and Symbols (points)
- *Data manipulation*: Create (new), Delete, Collect (select), Refresh (update the screen), Overlay (overlapping objects), Intersect (objects intersecting each other), Remove, Transform (Change the shape), Checkpoint, Rollback (history) and Commit.
- *Visualization*: Analysis, Interpretation

To develop an interface for human-GIS interaction it is necessary to use a systematic and formal language for characterizing and representing the analytical intents. In situations where complicated spatial analysis has to be performed, the aim is to guide and counsel the analyst and not to educate. To simplify the interaction between the user and the GIS tool (Krug, 2000)

(Cai, Yu & Chen, 2013), it must still be possible to execute functions matching the users intentions and plans, and to build the interface using a natural language.

> 'The rich application domains of geospatial analysis and the complexity of human conceptions of space mean that understanding the intentional structure of spatial analysis is likely to be challenging and involve interdisciplinary efforts.'
>
> (Cai, Yu & Chen, 2013)

To round up this section on HCI, we must also include the term usability. In general use-situations consist of three components (Cronholm, Goldkuhl & Ågerfalk, 1999); the user, a task and a tool. The strength of a system lies in the user and his/her task, which is to be able to match up the tasks with the appropriate tools.

### 4.2.2 Prospects of our mobile prototype

The physical and computational limitations of mobile devices are important considerations for applications intended for these kinds of devices (Peintner, Viappiani & Yorke - Smith, 2008). We must design an interface; which delivers the maximum benefits to the user (with only a small screen size), while keeping the user efforts to an acceptable level. As the screen sizes of tablets are small, in comparison to computer screens, and the use of fingers or pens being less precise than the cursor representing mouse, buttons needs to be larger and text to be more concise.

Tablet specifications in regards to the user interface

The interface skeleton below in Figure 4-2, matches (in the noted sizes) the screen size of an iPad, where the screen size is 19.7 cm x 14.7 cm.

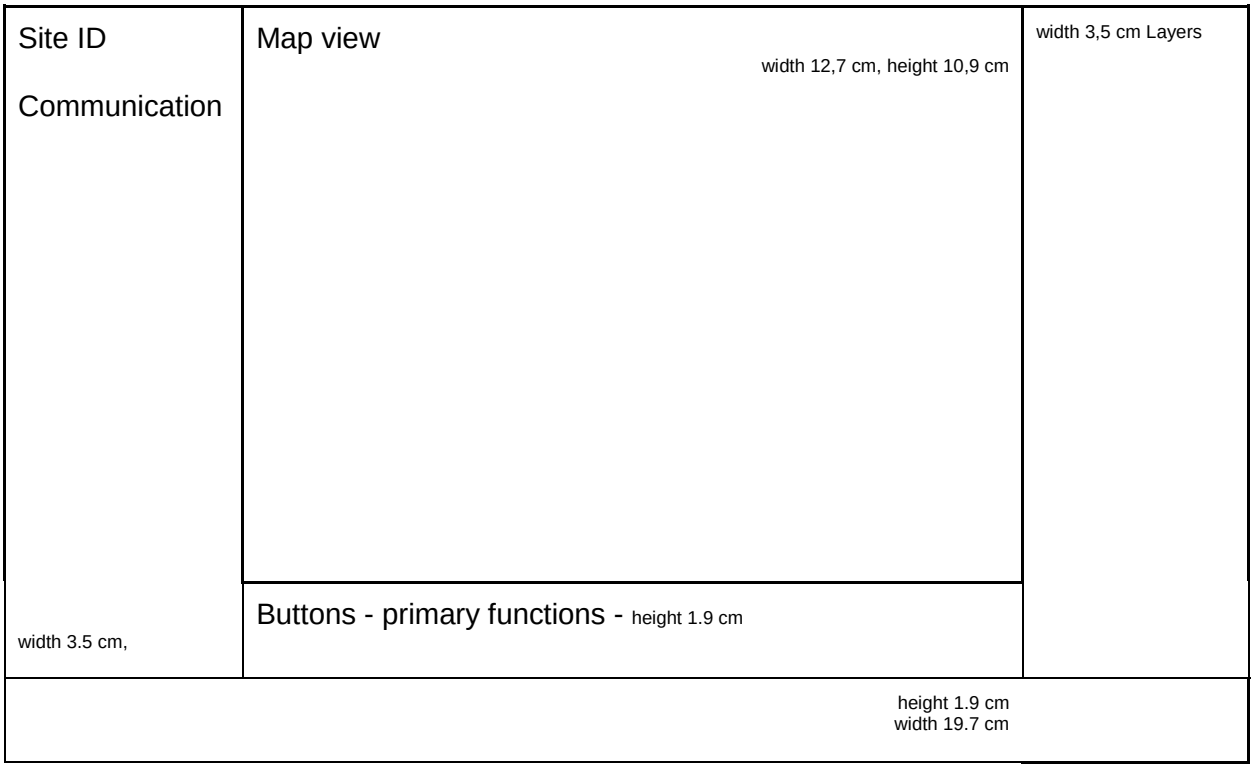



**Figure 4-2: Template of a user interface skeleton for an iPad (MTM group 4)**

An approximated size of the buttons for the iPad screen size is estimated to be approximately 1.5 cm x 1.5 cm or 2 cm x 2 cm, this should make it practical for even 'larger' fingers or glove covered hands to touch in the correct fields on the screen. It is also estimated that 4 mm is an appropriate distance between individual buttons. These sized leaves room for 6 to 7 buttons at the bottom of the page and 5 to 6 layers down the side of the screen.

Map appearance; Map view and the associated navigation tools

A geographical user interface should have a large map view and 2-4 navigation tools. The first priority will be two zoom buttons, zoom in and zoom out. The second and third priority in no particular order is a home button to ensure that the users location is centred in the map, and the other is a panning tool to move the to another map location.

Our background map must be selected with three basic principles in mind; simplicity (reducing noise), clear and adequate information (fulfilling the user's needs information) and recognizable features (making it possible for the users to identify their surroundings compared to the map).

Added Utilities, communication

A user interface should always have a utility to aid the users in times of need. The help to be offered; could be in the form of a communications field, information button, or maybe a text field with search possibilities. It should be possible for the users to get assistance on either software related questions and data related questions.

To facilitate communication concerning the system and to create an identifiable marker to be shared among the users, we have decided to name our software with a name which should be easy to remember; GERT. GERT stands for Geographic Emergency Rescue Tool, but it is also a Danish boy's name which will make it easier to remember. The identifiable marker GERT should also be included on the web site, increasing the users' familiarity with the name of the system.

Keeping the user interface simple and intuitive, while requiring as few clicks as possible for the users to gain their results; is our goal.

### 4.2.3 Prospects of Our Prototype for the NOST users

As the Danish NOST is the highest level of authority in connection with extreme emergency rescue operations, the staff will need more functions than the personnel in the field and at the same time they have access to larger screens, which gives us a possibility for including more functions in the user interface.

It is expected that the screen size is at least similar to a laptop, which is approximately 16.5 cm x 29 cm and that the NOST users will use both a keyboard and some type of mouse.

With regard to the functions of the program; NOST will need the similar functions that are included for the field officers, and furthermore they will need some more advanced functions e.g.:

- Route optimisation and calculations
- Printing options

- Logging of events
- Additional layers (which can be shared with the field officers if needed)
    - Height curves and blue spots maps, chemical factories, etc.
- Creation of flow direction layer
- Estimations of the water flow in connection with flooding (both the raising of water and the fall of the water level)

Many other functions could be required by the emergency rescue staff in connection with e.g. power lines, gas lines etc. (Cosic, Endersen, Foss & Palmqvist, 2013). We expect NOST operators to be responsible for all the urgent issues raised by the officers in the field.

For this project we only focus on a system relevant for a situation of flooding caused by extreme rain. A prototype for laptops and pc's should be similar to the tablet application, in regards to being intuitive and simple, and keeping the number of clicks low. But as the users have larger screens and more precise tools, as well as a mouse and a keyboard, there is a possibility for smaller buttons and incorporation of hotkeys. A laptop/pc application will have the room for smaller and more numerous buttons, and tabs and drop down listings.

## 4.3 Practical preparations to the development of our prototype

In order to have a functional solution we need to have a database to store information, a server to manage our stored data and a web server to display our stored data. At the client end (user interface) we need to have a web application and desktop program with a direct link to access the stored data from the web server, and to perform the required functions to assist and guide the emergency response teams. Since we already have a little knowledge of using GeoServer and PostGIS, we chose to use OpenGeo Suite. OpenGeo Suite is a full package solution that should enables all our requirements for displaying data to our prototype.

### 4.3.1 System architecture

The architecture of a system is defined by the structure and behaviour of the system. Figure 4-3 shows the program components and principles for interaction between the different program components.

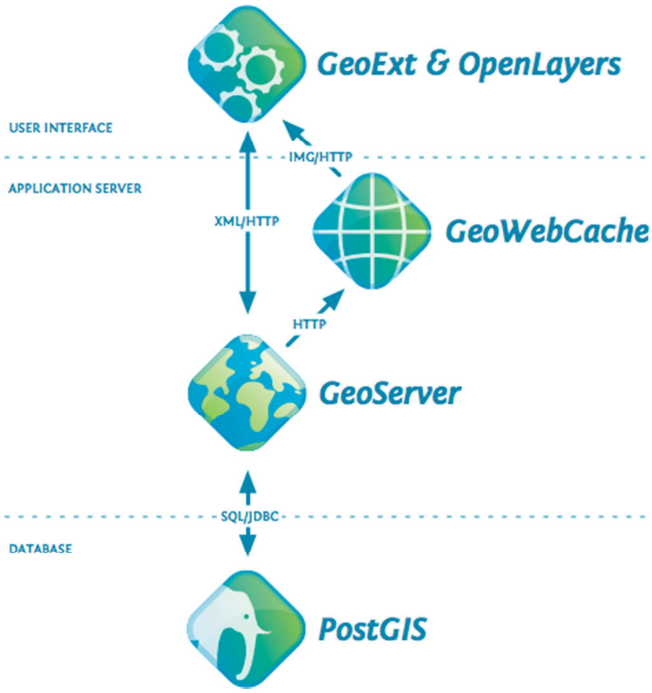



Figure 4-3: The architecture of OpenGeo Suite (Boundless Geo).[6]

The **PostGIS** database and application server GeoServer interact via Structured Query Languages (SQL) and Java DataBase Connectivity (JDBC)[7]. PostGIS is an extension to PostGreSQL, an object relational database that allows data and geographic objects to be stored. Based on the stored data, location queries can be run in SQL. Spatial objects like point, line, polygon, multi-point, multi-line string, multi-polygon etc. is expressed through WKT (Well-Known Text) form and WKB (Well-Known Binary) format. The GIS objects supported by PostGIS are 'Simple Features' defined by the OpenGIS Consortium (OGC) (OGC, 2011), Implementation and ISO Standard (ISO 19125).The database is programmed in Java like the GeoServer. The functions in PostGIS are primarily made in Procedural Language (PL)/PostgreSQL where SQL standards (full SQL92) are supported. The supported SQL commands can be seen at the homepage of PostGreSQL.[8]

Simple Features can be described through WKT and WKB, and include information about the type of the object and the coordinates which forms the object. In the PostGIS database; the stored geometry is saved in binary format e.g. 0100100000FD23533… and is therefore difficult to read (Obe & Hsu, 2011). WKT describes simple feature types and its orientation by coordinates, Figure 4-4 illustrates the WKT notations for a point, a line string and a polygon (OGC, 2011).

[6] http://boundlessgeo.com
[7] http://boundlessgeo.com/whitepaper/opengeo-architecture/
[8] http://www.postgresql.org/docs/8.3/static/features-sql-standard.html

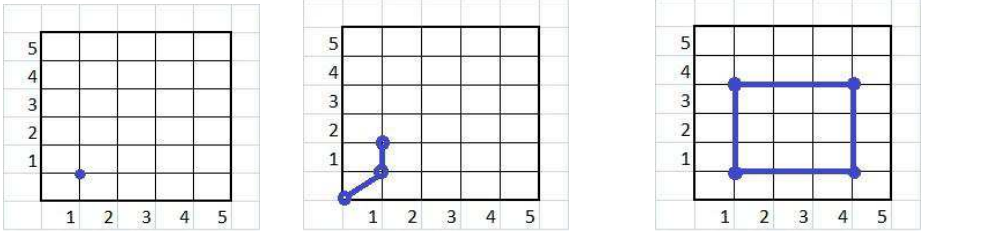



Figure 4-4: WKT examples (MTM group4).

Adding simple geometry data points to a new geometry point column in PostGIS can be done by SQL and built into functions in PostGIS:

1. WKT: POINT(1 1) and POINT (2 4) for input to my_table
2. TABLE: my_table(id serial NOT NULL PRIMARY KEY, name varchar(20));
3. SELECT AddGeometryColumn(schema_name, table_name, column_name, srid, type, dimension);
4. SELECT AddGeometryColumn( 'public', 'my_table' , new_points, -1, 'POINT',2);
5. INSERT INTO my_table (name, new_points) VALUES ('first', ST_GeomFromText ('POINT(1 1)')); INSERT INTO my_table (name, new_points) VALUES ('second', ST_GeomFromText ('POINT(2 4)'));

Line 1+2 defines two points for input to my_table, where the table consist of two columns 'id' and 'name'. Line 3 shows the default syntax for adding a geometry column with the AddGeometryColumn function, and line 4 the syntax for our example. Line 5 is where the points from line 1 are put into new_points and 'point name' into name.

**GeoServer** is an application server that allows users to share and edit geospatial data through web services. It's structure follows OGC reference standards for WMS (Web Map Services), WFS (Web Feature Service), WCS (Web Coverage Service), and support interoperability by being able to publish data from any major spatial data source like: PostGIS, Shapefile, ArcSDE (Arc Spatial Database Engine), DB2 (Database 2) and Oracle in output formats like: GML (Geography Markup Language), CSV (Comma-Separated Values), JPEG (Joint Photographic Expert Group), GIF (Graphics Interchange Format), PNG (Portable Network Graphics), PDF, SVG (Scalable Vector Graphics), KML (Keyhole Markup Language), GeoRSS (Geo Rich Site Summary). A full list of features can be seen at the GeoServer web page.[9] The application server and user interface layers interact via standard web encodings (XML (Extensible Markup Language), JSON (JavaScript Object Notation), images) over a HyperText Transfer Protocol (HTTP) transport.

**GeoWebCache** is a web application that delivers map tiles, e.g. geospatial data, fast from OGC Web Map Service standard servers using a cache (a store of tiles made in advance). GeoWebCache is different as it operates against any compliant Web Map Service (WMS) implementation. GeoWebCache acts as a proxy between the client and the WMS server. When a client requests a tile, the program will first check whether this tile has been requested before. If a tile matching the parameters has been stored, then it will simply return that. In this case, the tile cache will often be one hundred times faster than a WMS server, because all it has to do is stream a file to the network.

OpenGeo supports the OpenLayers JavaScript map library and the GeoExt JavaScript user interface library. OpenLayers is an Open source library; that makes it easy to load a dynamic

[9] http://GeoServer.org/display/GEOS/Features

map in any web page. It can display map tiles and markers loaded from any source and display the data in a web browser. GeoExt is a compatible extension to OpenLayers that will allow building more complex GIS applications on the web with JavaScript.

### 4.3.2 Input data

Here we refer to the WMS and WFS standard protocols used by servers; to exhibit and make georeferenced data accessible over the web from databases.

#### Web Map Services (WMS)

WMS is a standard protocol by Open Geospatial Consortium[10] for serving georeferenced map images which can be retrieved by the internet through a simple HTTP interface from one or more geodatabases. There are several versions; the 1.3.0 standard is the most recent OGC specification (OGC, 2006). GeoServer supports WMS in version 1.3.0 and 1.1.1. The major difference between the two versions; is that the axis order for the geographic coordinate defined in the EPSG (European Petroleum Survey Group) database has been reversed.

WMS defines the geographic area of interest and the layers which should be retrieved from the geodatabase. The defined geographic area will be returned as one or more geo-registered map images in JPEG, PNG, and etc. format. These images can be displayed in a browser application; images can also be specified to be returned as transparent. WMS services use the following operations:

- *GetCapabilities*: A request to a WMS server to get information about the server identification, operations and services such as supported geographical reference systems and boundary box; which are offered by the server.
- *Get Feature Info*: This operation gets the actual data such as attribute values and geometry for a pixel location.
- *Getmap*: A request from the client to retrieve the actual image. After getting information about the server in GetCapabilities, the client can specify the boundary box, a spatial reference system, width and height, and style and format of the image.

#### Web Feature Service (WFS) and Web Feature Service Transaction (WFS-T)

Web Feature Service Interface standard (WFS[11]) is an Open Geospatial Consortium standard (OGC, 2010), which provides an interface for retrieving and updating the geographical features; written in Geography Markup Language (GML[12]) a subset of XML (OGC, 2007) across the web from multiple web feature services. Depending on which version of WFS is used, the available operations vary. All WFS services support the following operations; GetCapabilities, DescribeFeatureType, GetFeature, Transaction and Lock Feature:

- *GetCapability:* It is important to the client, who connects to a WFS server; to know which functions (capabilities) are available on a specific server. The GetCapability functionality returns a XML document; which contain five main components; Service Identification, Service Provider, Operations Metadata, Feature Type List and Filter_Capabilities.

[10] http://www.opengeospatial.org/standards/wms
[11] http://www.opengeospatial.org/standards/wfs
[12] http://www.opengeospatial.org/standards/gml

- *DescribeFeatureType:* Returns the field information about one or more dataset supported by a WFS service. Useful when requiring more information on attributes of one or more feature types.

- *GetFeature:* Asking for and receiving main map data. Using GetFeature can result in receiving enormous amounts of data, but it is also possible to specify the exact data of interest to be retrieved from the server. This will help for limiting the output and returning a selection of features from a data source.

- *Transaction:* Features published by a WFS server can be modified, edited and deleted. The WFS server which supports the Transaction is called WFS-T. By defining the action area it is possible to pull out all the relevant data from the area of interest, and at the same time modify, edit, delete or add a new feature.

- *LockFeature:* This function aids to ensure consistency and serialization throughout the editing of transactions. If a user starts to modify a feature, LockFeature will prevent other clients in editing the same feature at the same time, before submitting it back to the WFS. In the server GetCapabilities response it is possible to see if the WFS server supports LockFeature or not.

The following operations are available only in version 2.0.0:

- *GetPropertyValue:* When using query expressions it is possible to retrieve a part of or the whole property value of a feature.
- *GetFeatureWithLock:* A selection of locked features is returned.
- *CreateStoredQuery:* Saving queries on the WFS server.
- *DropStoredQuery:* Delete the stored queries.
- *ListStoredQueries:* Returns a list of stored queries
- *DescribeStoredQueries:* Returns metadata on the stored queries.

To sum up the use of online data, WMS provides access to raster layers. WFS and WFS-T provides access to vector layers such as polygons, polylines and points, the types of data served as WFS are e.g. roads, building and points of interest. WFS-T is used for data; which is to be available online and at the same time to be modifiable.

#### 4.3.3 Metadata

Metadata is the description of data, and what is known about datasets. There are two main purposes of metadata:

A. Metadata gives information about the data; for example how, when and by whom the data was created. This information gives confidence to the data user. No one would accept data for serious work without some information describing the data.

B. Metadata makes data searchable by keywords. Thematic, geographic and temporal keywords make it possible to find the dataset among large amount of spatial data.

The Metadata structure should always be defined by common standards for easier searching and comparison of data. There are two international ISO (International Organization for Standardization) standards for metadata as well many other national standards around the

world. Most of these standards have a great deal in common with each other. It is the result of many international conferences and meetings, that we have the common ISO standard today.[13]

- ISO 19115: is an ISO standard which published an approved in 2003[14]
- ISO 19119: is an ISO standard metadata for applications[15]

The abstracts of the standards ISO 19115 and 19119 from ISO can be seen in appendix 6.

In Denmark we follow the INSPIRE (INfrastructure for SPatial InfoRmation in Europe) specification for metadata as a standard. A web portal for metadata has been set up by GST, where it is possible to search for spatial datasets and services, in accordance with the INSPIRE directive.[16]

Common infrastructures for geodata in the European countries will be reached by using INSPIRE. The main objective for INSPIRE is that geodata can be used locally, nationally and across the borders between the European countries. The types of geodata to be covered by the INSPIRE directive is to cover the areas of e.g. the environment, transportation, agriculture, health, etc. Another reason is ensuring reuse of the same data regardless of primary intensions to the use of the data. The intention is that time and money can be saved not doing the same data collections and data manipulation twice. In Denmark GST is responsible for the implementation of INSPIRE.

#### 4.3.4 Programming

HyperText Markup Language (HTML) today is used in almost all internet sites. The language makes it possible to embed images, objects and other scripting languages on internet sites, e.g. JavaScript. HTML can also be used to define an internet page's appearance, but this is most often being left to the Cascading Style Sheet (CSS) which is a style sheet language used for describing the presentation semantics.

HyperText Markup Language (HTML)

Hypertext Markup Language (HTML) is a markup language that browsers use to interpret and compose text, images and other material into visual or audible web pages. HTML was developed by two CERN (The European Organization for Nuclear Research) scientists in Switzerland, with Tim Berners-Lee as the main architect. The purpose of the language was to give scientists a simple markup Language, to present their scientific results on the Internet. HTML was first published as an Internet draft in 1993 and during the '90s there was the huge activity around HTML with version 2.0, version 3.2, version 4.0, and finally 1999 version 4.01. HTML5 is the latest revision of the HTML standard and is currently the working draft of the World Wide Web Consortium (W3C). HTML5 adds changes to the HTML language, especially several new syntax features such as <video>, <audio> and <canvas> elements as well as the integration of scalable vector graphics.

[13] http://www.iso.org/iso/home.html
[14] http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=39229
[15] http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=39890
[16] http://geodata-info.dk/Portal/

Wanting to take the web platform to a new level, a small group of people started the Web Hypertext Application Working Group (WHATWG) in 2004. They created the HTML5 specification. They also began working on new features specifically geared to web applications; the area which they felt was lacking the most. It was around this time that the term Web 2.0 was coined. And it really was like a second new web, as static web sites gave way to more dynamic and social sites that required more features, a lot more features (Lubbers, Albers, Salim, 2010).

HTML5 introduces many new markup elements, which is grouped into seven different content types. These are shown below in Table 4-1.

| Content Type | Description |
|---|---|
| Embedded | Content that imports other resources into the document, for example audio, video, canvas, and iframe |
| Flow | Elements used in the body of documents and applications, for example form, h1, and small |
| Heading | Section headers, for example h1, h2, and hgroup |
| Interactive | Content that users interact with, for example audio or video controls, button, and text area |
| Metadata | Elements—commonly found in the head section— that set up the presentation or behaviour of the rest of the document, for example script, style, and title |
| Phrasing | Text and text markup elements, for example mark, kbd, sub, and sup |
| Sectioning | Elements that define sections in the document, for example article, aside, and title |

Table 4-1: HTML5 content type (Lubbers, Albers, Salim, 2010)

One content type that contains many new HTML5 elements is the sectioning content type and these are shown below in Table 4-2.

| Sectioning Element | Description |
|---|---|
| header | Header content (for a page or a section of the page) |
| footer | Footer content (for a page or a section of the page) |
| section | A section in a web page |
| article | Independent article content |
| aside | Related content or pull quotes |
| nav | Navigational aids |

Table 4-2: New sectioning HTML5 elements (Lubbers, Albers, Salim, 2010)

The most basic HTML document consists of three elements. The first elements are <html> and </html> between them are various item attributes as well as text or graphic content located. A basic HTML document might look like the source code below:

```
<!DOCTYPE html>
<html>
   <body>
          <h1>'MTM Gruppe 4'</h1>
   </body>
</html>
```

Source code X. Basic HTML syntax

HTML Elements comes usually by tag pairs.

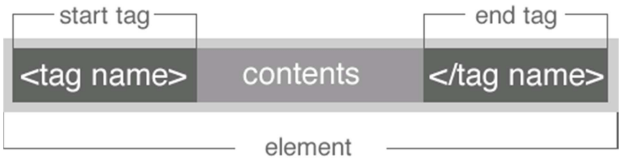



**Figure 4-5: HTML element (w3.org)**[17]

For opening a simple element with a start tag

1. it starts with **<**
2. then a list of characters without space, the tagname (or element)
3. ends usually with a **>**

Then closing the simple element with an end tag

1. it starts with </
2. then the same list of characters without space, the tagname (or element)
3. it ends usually with a **>** (World Wide Web Consortium, 2013 ).

### Cascading Style Sheet (CSS)

Cascading Style Sheets, also called CSS is a computer language used to describe how you want the content of an HTML / XHTML (Extensible HyperText Markup Language) / XML document presented in a browser. You can embed CSS language in an HTML document between two style tags:

```
<style type='text/css'>
                #map { height: 320px; margin: -15px;}
                body {
                        padding: 0;
                        margin: 0;
                }
                html, body, {
                        height: 100%;
                }
```

[17] http://www.w3.org/wiki/HTML/Training/Tag_syntax

```
</style>
```

or have it in a separate file by reference via a link tag:

```
<link rel='stylesheet' href='basis.css' type='text/css' />
```

The advantage of placing this type of information in a separate file is that the formatting can be reused in multiple documents, as multiple documents can include links to the same CSS file.

And finally, it is possible to write CSS directly in an HTML element:

```
<p style='color: red; margin-left: 15px'>
Hello World!
</p>
```

CSS3 is based on the widely used CSS 2.1 standard and is the W3C latest revision of the CSS standard and the current Working Draft. Several of the advantages associated with HTML5, is actually also associated with CSS3 and JavaScript.

### JavaScript and ECMAScript

JavaScript is an interpreted programming language with object-oriented (OO) capabilities, and it draws inspiration from the Perl programming language in a number of areas, such as its regular expressions and array handling features (D. Flanagan, 2006).

JavaScript does not have the same properties as traditional programming languages, for example it is not possible to write a program directly in the computer control system using JavaScript, as it is possible with Java, C++ or NET.

ECMAScript is the scripting language that forms the basis of JavaScript. ECMAScript is standardized by the ECMA International standard organisation in the ECMA-262 and ECMA-402 specifications.[18]

JavaScript is implemented in two ways in an HTML document either directly in the HTML code, see code below:

```
<script type="text/javascript">
document.write("Hello World!");
</script>
```

Or as a link in HTML <head> tag, see code below:

```
<head>
<script type="text/javascript" src="script.js"></script>
</head>
```

JavaScript libraries are libraries of existing JavaScript; which developers can call back and recycle in JavaScript based applications. This makes it easier for developers to create JavaScript (JS) applications, as it is not necessary to rewrite everything. For the development of

[18] https://developer.mozilla.org/en-US/docs/JavaScript/Language_Resources?redirect=no

web applications there are also specific JavaScript (JS) libraries, for example, Leaflet, Jquery Mobile and Sencha Touch respectively, Jquery and Ext Core.

## 4.4 Developing a prototype tool for a rescue and warning system

When creating a rescue and warning system, one of the main issues to keep in mind is being able to locate both the user, other users, and obstacles with great precision. Therefore we will touch on the subject of geolocation and geographic positioning systems in section 4.4.1, and include a section on the limitations of identifying night and day residents of local areas. In Denmark there are many regulations describing dos and don'ts in connection with the monitoring of the population, where they live and who they are.

### 4.4.1 Geolocation and the global positioning systems

Geolocation is the designation of the geographic position of a person, object or place. Today the Global Positioning System (GPS) is the main tool used for positioning. In this section we name and describe the geolocation methods of our modern society.

#### Geolocation methods

There are many ways for modern computing to find their geolocation. See the following:

- Global Positioning System
- IP Addresses, WiFi and Bluetooth MAC Address
- Cell IDs

#### Global Positioning System (GPS)

We will briefly introduce the history and technical foundation of GPS and other navigation systems; which are currently in use or under construction, resembling the growing market of navigation systems and the available opportunities.

GPS- Global Position System is a worldwide system to determine positioning. There has been military investment behind the development of GPS. The U.S. Department of defence started to develop the GPS system in 1973, under name of Navigation Satellite Timing and Ranging (NAVSTAR). The GPS system was eventually fully operational on the 8th December 1993 with 24 satellites positioned in space. Early on it was possible to see the use of civilian applications. Even today the GPS system continues to grow, developing existing systems and improving coverage and accuracy. [19]The military interests behind the development of GPS originally included a feature called Selective Availability. Selective Availability is a feature which intentionally adds time varying errors, up to 100 meters. This was disabled from the year 2000 on the order of President Clinton. Selective availability can be turned on again on at any time; this has to some extent urged other governments into developing their own positioning systems (Dueholm & Laurentzius, 2002).

Russia has a system called GLONASS (GLobal NAvigation Satellite System), at this point in time it is the only alternative navigational system in operation with global coverage and

[19] http://www.gps.gov

comparable precision. Many new devices support both GPS and GLONASS; this gives the opportunity to access more satellites, resulting in better coverage and better results creating more confidence in the systems.

Until now the global navigation satellite users have been dependent on GPS and GLONASS, and therefore the European Union (EU) are developing their own system called Galileo. Galileo will, when completed, contain 30 satellites (27 operational and 3 active spares) in 3 orbits at an attitude of 23.222 km. Galileo should be fully operational by 2019 (Figure 4-6).

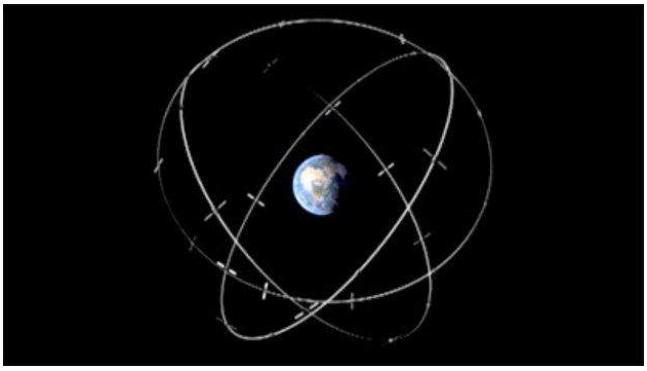


Figure 4-6: Illustrates the constellation og Galileo satellite system (European Space Agency (ESA)[20])

These 3 orbit planes inclined at an angle of 56 degree to the equator, unlike the GPS and GLONASS, Galileo is characterized as a civil system. Galileo will give better accuracy for measurements and positioning (within one meter precision for both horizontal and vertical), it will be fully interoperable with GPS and GLONASS. Galileo will include applications to facilitate different areas of use. As for example agricultural, transport, fisheries, science- environment, and weather and rescue operations which is relevant to our research and are defined as the following in the European Global Navigation Satellite System (GNSS) Agency:

- Galileo-ready devices will enable new security-related applications, permitting the location of stolen property, for example, or lost pets or individuals.
- Galileo signals facilitate civil protection operations in harsh environments, speed up rescue operations for people in distress, and provide tools for coastguards and border control authorities.
- Galileo's Search and Rescue (SAR) function is linked to the operational Cospas-Sarsat system, wherein satellites are equipped with transponders that can transfer distress signals from user transmitters to a rescue coordination centre.

European GNSS Agency[21]

In the future there will be more satellite systems available to ensure better positioning and geolocation on earth. For information on how GPS works, please see appendix 7.

[20] http://www.esa.int/
[21] http://www.gsa.europa.eu/galileo/applications

### Source of errors in GPS

There are many sources of errors which can caused by the lack of accuracy and even availability of GPS satellite signals, some of these errors can be reduced or be eliminated by GPS receivers, and some can still be challenging to the users. In this report we include some of common error sources which GPS users should know about; satellite geometry, multipath and atmospheric effects.

### Satellite geometry in general

Triangulation is used for calculating positions. Small errors in the distance measurements can cause large errors and inaccuracy in the resulting position. Satellite geometry concerns the position of satellites in relation each other from the view of the receiver.[22] When satellites are close together (in same direction from the point of view of the receiver), the accuracy of the position is low. In the worst case no location is found. (Dueholm & Laurentzius, 2002)The best accuracy can be achieved by receiving signals from many widely dispersed satellites (Figure 4-7) at the same point in time.

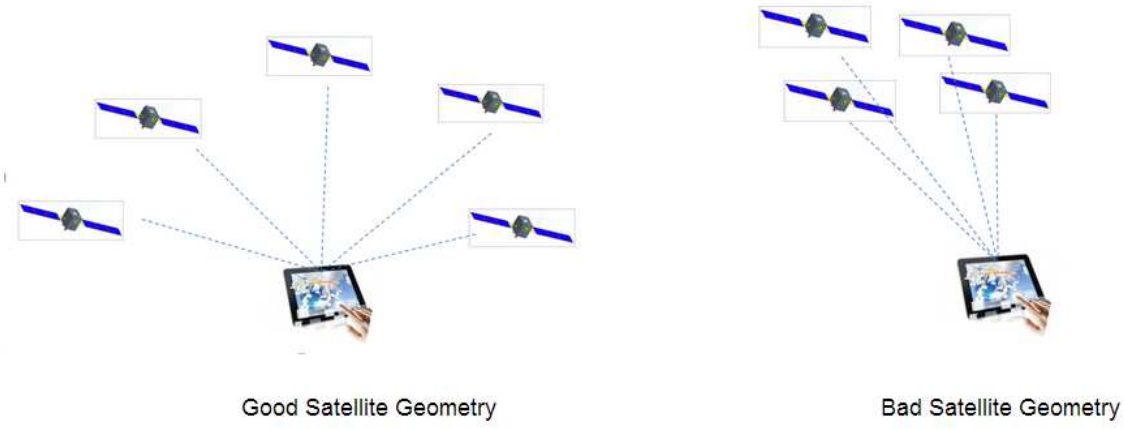



**Figure 4-7: Illustrates the effect of GPS constellations (MTM group 4)**

### Multipath

The multipath effect can occur in situations where the satellite signals are reflected on e.g. high buildings and other objects. The signal reflected on objects take a longer time to reach the receiver than a direct signal. (Dueholm & Laurentzius, 2002) The prolonged path of multipath signals can cause inaccuracies of receivers in the range of a few meters. Please see Figure 4-8; which illustrates the prolonged paths of multipath signals.

[22] http://extension.missouri.edu/explorepdf/envqual/wq0452.pdf

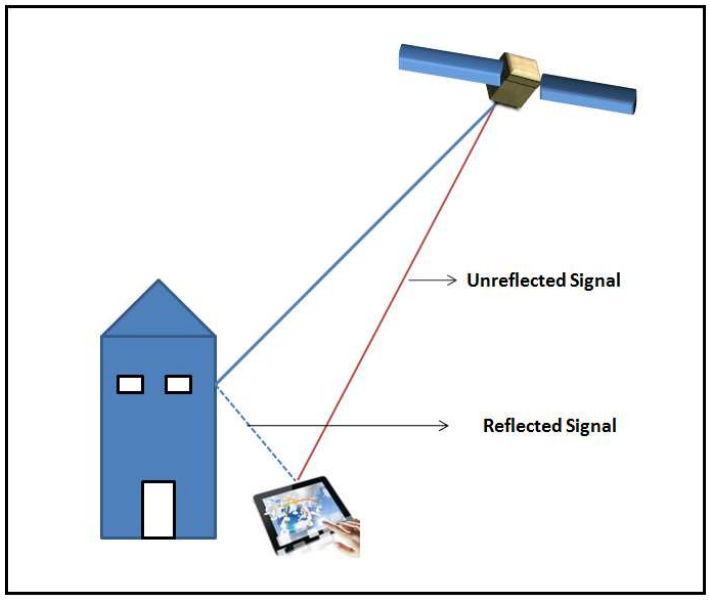



Figure 4-8: Illustrates the prolonged path for satellite signals called multipath (MTM group 4)

Atmospheric effects

The troposphere and ionosphere play a large part in the inaccuracies of finding the exact location of a receiver. The velocity of GPS signals travel through the atmosphere is reduced by the atmospheric effects. The speed of GPS signals is slower in the ionosphere and troposphere than in the outer space. The troposphere is the part of Atmosphere which is closest to the earth's surface. By predefinition of elevation mask on the device a large effect of troposphere can be avoided.[23] The delay of the signal in the troposphere depends on the temperature, pressure and humidity. Conditions at the earth's surface are rarely the same as those higher up in the troposphere. The ionosphere is the part of the atmosphere which lies approximately 50 km from the Earth's surface and up to around 1500 km high. The cosmic rays and the sun's ultraviolet radiation cause the air molecules to split into ions and electrons. The density of the free electrons varies on the time of day, season and sunspot activity over a number of years (Dueholm & Laurentzius, 2002).

### 4.4.2 IP Address, WiFi and Bluetooth

IP (Internet Protocol) Address is a unique numeric identifier for every device; which is connected to the internet. Just like we use our home address to send and receive post, computers and all other devices which are connected to the internet use IP addresses to communicate with a specific computer on a network. Webmasters can use IP to trace the visitors of their sites, advanced web pages can change the content of their site based on the physical location of the visitors. It is also possible to block access to a web page from certain countries or local IP addresses. An IP address might look like this: 95.166.187.133. It contains four numbers separated by a dot (.), these four numbers can range from one to three digits and each of these numbers can range from 0 to 255. This unique number is a key to sending and retrieving data by the internet. The location of a billion devices; connected to the internet can be pinpointed by using the IP addresses gathered from the devices. An IP address can be static (permanent) or

[23] http://www.csr.utexas.edu/texas_pwv/midterm/gabor/gps.html

dynamic (temporary) and in both cases they are assigned with a specific Internet service provider within geographic blocks. The geographic blocks are based on a region, this is set by a local registry institution, that is why it is easy to identify the country, region and even city by use of IP addresses.

There are two big challenges for identifying geolocation by an IP address. The first challenge is that there are many of regional institutions which should be required to obtain the data. The second challenge is that the location identified by the IP address can be very misleading, because the location of the ISP (Internet Service Provider), firewalls or any other devices that passes the data to the device; which we are interested to get the geolocation of. This method is mostly used by business owners who want to know more about their potential customers and analysing their market data.

To Geolocate a device; by use of WiFi and Bluetooth MAC addresses, is similar to geolocation by IP addresses. Each device has a MAC (Media ACcess address) which is a unique number dedicated to the interface by the manufacturer of the interface card. In modern devices it is allowed to change the MAC address manually despite of the primary idea; that the address should be permanent and globally unique. By using MAC address like using IP address it will be possible to obtain both latitude and longitude of a device.

### 4.4.3 Cell IDs

The *Cell ID i*s a unique number for identifying mobile phones and other devices in a particular cell network regardless of what type of technology the devices are using. Mobile devices communicate with Base Transceiver Station (BTS). Many telecommunication companies place more antennas on every mast Figure 4-9 to cover all directions. [24]

**Figure 4-9: shows how the telecommunication companies cover all directions by putting more antennas on one BTS (mobilsiden.dk)[25]**

Each BTS sends the Cell Id to its cells and mobile devices; so that they can always receive messages. The mobile devices can calculate their approximate location by knowing their Cell ID and the geographic location of the corresponding BTS (Trevisani E. & Vitaletti A, 2004). To determine a location of a device by Cell ID, the device should have access to at least 3 BTS,

[24] http://www.teleindu.dk/wp-content/uploads/2012/03/brochure1_final_print_18062013.pdf
[25] http://www.mobilsiden.dk/tips/sadan-booster-du-hastigheden-pa-mobilt-bredband,lid.21401/

more towers gives better accuracy and that is why the accuracy in urban locations are higher than at rural locations. The triangulation principle allows us to determine the geographical location of a device. (Holdener T.A, 2011)

The principle is simple, when a device gets signals from 3 towers, the device can be identified by finding a common conjunction point between these three signals, in the figure below the receiver can only be in one place, Figure 4-10.

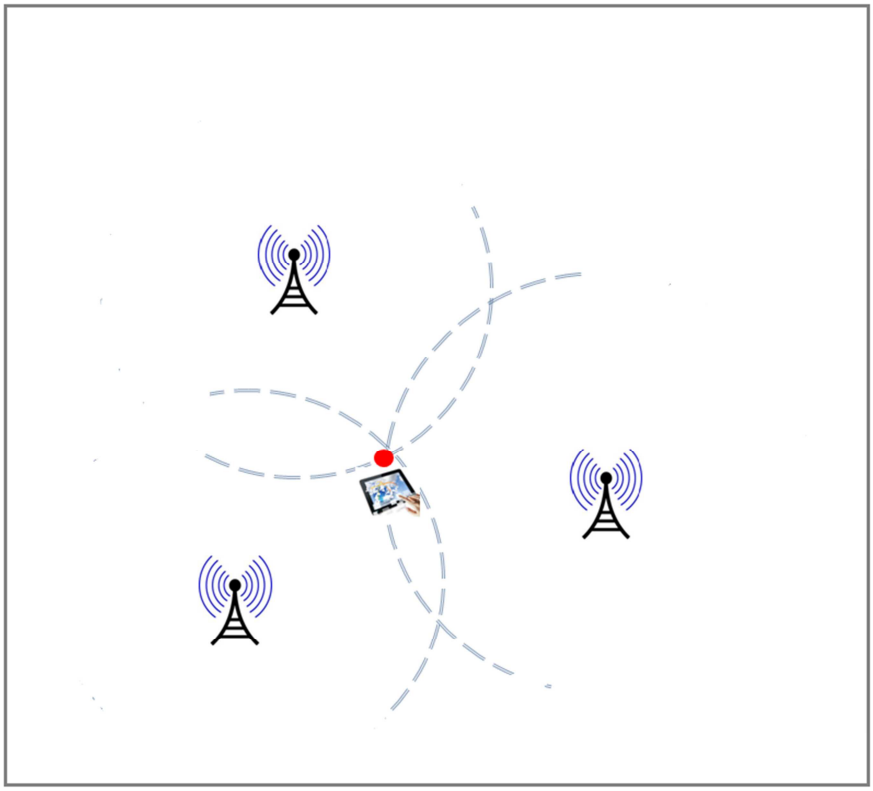


Figure 4-10: Illustration of triangulation method (MTM group 4)

In this section we described the different ways of geolocation. Each of these methods has their advantages and disadvantages. Despite of multipath and bad geometric constellations; which can lead to increased accuracy, geolocation by satellite systems are the most precise, but the biggest challenge is not getting access to sufficient satellites in order to determine a position in urban areas with high buildings. The following preferable method to ensure accuracy is geolocation by Cell ID; which requires access to at least 3 BTS. Geolocation by IP address, WiFi and Bluetooth is less accurate and the accuracy can be offset by kilometres, the location derived from IP address can be misleading because of the ISP (Internet Service Provider), firewall or other devices.

## 4.5 Software Testing

As this is a project involving an actual piece of software, testing must be included. How else will we be able to follow our own progress? The purpose of testing is dependent on the project type. When a project requirement specification exists, all of the listed requirements must be met.

Testing from an early stage of the project, ongoing until the end, will help reduce the total amount of errors. In this way errors will be found, described and corrected as soon as possible. Documenting our errors and corrections will also help us keep trace on our project.

> 'An error occurs, when software does not behave in the way that a user, within reason, can expect.'
>
> (Olsen & Vinje, 2004)

A well planned test must be both efficient and effective as described in the book 'Softwaretest – Kom godt i gang' (Olsen & Vinje, 2004).

*Effective: finding errors*
*Efficient: keeping the price per error low (not too much re-work)*

> 'The test must document the existence of the agreed function and visualise that it is correct. The test must also identify the difference between the expected and the actual result, in this way the test assists in finding errors.'
>
> (Olsen & Vinje, 2004)

As our project is focused on development, the time estimate of testing during this entire project should be set at 25-30% (Olsen & Vinje, 2004), which is the normal rate of time used for testing in a project of development. The goals of our tests can be view as administrative and information/service (Olsen & Vinje, 2004). In this case it is software directly intended for the emergency response teams in Denmark and the testing would also be critical. The functions which we have selected for our project reflect the field officers' needs for documenting and assessing a situation of flooding, therefore we also address important life critical issues.

Life critical:

- Risk assessment: A field officer must be able to assess the risk of flooding and the loss of life risk and material values in the area affected.
- Evacuation: Should the public be evacuated? If so; where from, where to and along which roads.
- Prevention: Can anything be done to stop the water? Can the water be re-routed?
- Action: Identifying the primary target area of the operation, secondary target area etc.

Administrative:

- Logging; for later evaluation
- Communication
    - To the NOST
    - To the fellow field officers

Information/service:

- Warning system
- Communication
    - To the media
    - To the public (web pages, email, SMS (Short Message Service))

> ‘For life-and-death situations, you want exhaustive, carefully designed, quantitative, large-sample, reproducible, scientific studies that produce statistically significant results. Or at least I do.’
>
> Steve Krug (Krug, 2010)

Software testing is a combination of verification and validation. Verification concerns whether it is the correct product; which has been produced for this specific purpose. This should be tested during the product development. Validation is a process to ensure that the product does what it is supposed to do.

Next is described the use of white-box testing and black-box testing, and a bit of grey-box testing. Here; a short introduction to some of the differences between white-box testing and black-box testing.

White-box Testing
- Positive approach, everything is expected to work properly
- Covers all functions
- Does not cover code strings which are without function.

Black-box Testing
- Negative approach, the purpose is to identify errors
- Everything is tested, both function- and background code
- Cover the client’s business needs
- Involves both system testing and user tests

Positive testing: What can this application do? Perform testing with valid inputs.
Negative testing: What is this application ‘not’ supposed to do? Perform testing with invalid inputs.

#### 4.5.1 White-box tests - the 0-error testing.

The 0-error testing is the programmer’s own testing of functions, including input data and output data. Test is performed simultaneously with the progress of the project. This is a theoretically correct model (Olsen & Vinje, 2004). But in practice more problematic. Tests needs to be run and re-run of the entire system even before the project is completed. Acceptance will have to be founded on the requirements specification. User testing must be based on an analysis report, the system test on a prototype and so on.

> ''Do-it-yourself' usability tests are definitely qualitative. The purpose isn’t to *prove* anything; it’s to get insights that enable you to *im*prove what you’re building.'
>
> Steve Krug (Krug, 2010)

The Radical Test Model is based on the principles of 0 errors in development. No flaws will pass through a phase during development. The system is in principle flawless at any given point in time of the development. In practice the developer is to test all functions; before releasing them for actual testing. This matches well with the principles of primary testing using white-box testing techniques. The white-box testing techniques involve testing the programming, separating

individual pieces of text strings and making sure they work perfectly. Later black-box testing will be predominant. black-box testing involves testing the overall image, without access or knowledge of the programming which lies behind.

The mixture of white-box testing and black-box testing can be called grey-box testing. This leads to the next section Usability testing, which can be part of black-box testing.

### 4.5.2 Usability testing and black-box testing

This can be tested by others than the programmers themselves. Black-box testing is having someone other than the programmer himself test the software or function. It should be considered whether to focus on testers sitting alone, maybe using a manual, or if there is time (and money) to observe and guide testers using software and functions.

> 'Watching people try to use what you're creating/designing/building (or something you've already created/designed/built), with the intention of (a) making it easier for people to use or (b) proving that it is easy to use.'
>
> Steve Krug (Krug, 2010)

One of the differences between white-box testing and black-box testing is testing the non-functional parts of an application; this includes the user interface and GUI testing. Validation of whether the user interface is professionally designed for the expected users or not. Usability testing includes testing the user friendliness, comprehension and use.

Security testing would in the case of an actual application for the DERS be of high importance. Security testing is a validation of whether all security conditions are complied. Data security is an evaluation of whether the needed data is available, if the data is informative, adequate and precise. Can the data and software suppliers be trusted? Can the software be trusted? How stable is the server and can the system be hacked? How often are the system and the data updated? How stable is the system?

Testing the settings, like time and language, of the system is part of Performance testing; this also includes installation, and compatibility towards other software products (e.g. accessing the databases in for example ArcGIS), etc.

### 4.5.3 Quantitative tests and qualitative self-testing

Proving improvements; by measurements as success rates and time consumption (Krug, 2010). Quantitative tests are like scientific experiments: a test protocol is needed and all participants must follow it. In qualitative testing we will need several testers, doing the exact same actions and observe their results without interference. For the purpose of gaining insights that can enable us to improve our application (Krug, 2010). During qualitative testing we can observe, advice and adjust the task and the protocol to fit our purpose of gaining knowledge.

To achieve a complete testing coverage is close to impossible. But by performing qualitative self-testing and requiring a tester or two to go through the functions of our program will bring us closer to an acceptable quality level. Time and money are the main issues as to why quantitative testing is often neglected.

'What testing can do is provide you with invaluable input which, taken together with your experience, professional judgement, and common sense, will make it easier for you to choose wisely—and with greater confidence—between 'a' and 'b.''

'Testing is an iterative process. Testing isn't something you do once. You make something, test it, fix it, and test it again.'

Steve Krug (Krug, 2000)

#### 4.5.4 Testers from outside the project and exploratory testing

Usability testing with testers without prior involvement in the project, but still with an interest in the result and future use of the software, can be a great bonus to the program. For example to invite personal from the DERS teams to test our software or as an alternative ask, other similar and qualified people to test, e.g. fellow students or unbiased programmers.

'Try to find users who reflect your audience, but don't get hung up about it.'

- 'Get it' testing is just what it sounds like: show them the site, and see if they get it—do they understand the purpose of the site, the value proposition, how it's organized, how it works, and so on.
- Key task testing means asking the user to do something, then watching how when they do.

'Your responsibility is to tell the users what you want them to do, to encourage them to think out loud, to listen carefully to what they have to say, and to protect them' (See appendix 8 for a list of pointers to remember before testing).

Steve Krug (Krug, 2000)

Using testers can be very time consuming (and costly), therefore not everyone uses it. For us, it would be a very good indication of whether or not we match the correct target group and their primary wishes and demands. It is important to factor in what is achievable in the time allowed and that may mean the use of testers in a certain situation is not practical. Many details have to be taken into account.

- Instructing the testers on the program functions.
- Watching the testers perform tests and taking notes.
- Evaluating the test performance together with the tester.
- Final evaluation and comparison of test results for all tests.

*'*As soon as possible after the test, each observer and the facilitator should type up a short list of the main problems they saw and any thoughts they have about how to fix them.'

(Krug, 2000)

Some requirements of operation would have to be a certain level of technical skills and insights to the expected use of the application by the planned users.

As a final test of the application, testing has to be performed on the non-functional parts. For example validating the user interface: Is it a professional design or not? A checklist of some of the non-functional elements to be tested can be seen in Appendix 9.

### Exploratory testing

*'Exploratory testing is simultaneous test design, execution and learning'* (Kohl, 2013). The perspective of exploratory testing is that it is human centred (the tester is not dehumanized by reducing the testing to a clerical task or by trying to automate the testers). The unique skills and experiences of testers are sought to add value to the project. Exploratory testing does not have a clear structure, there are no explicit steps to follow and expected results to be meet. The test executing differs depending on who is doing the testing. There are several styles and models for exploratory testing; e.g. Intuitive, systematic and regression styles. A more in-depth description of the styles can be found in Appendix 10. Exploratory can be a disciplined, structured form of testing.

> 'Exploratory testing isn't just 'playing with a product until it breaks,' and it doesn't need to be a set of simple 'quick tests.' Often improperly characterized as 'superficial bug hunting' or 'quicktests', exploratory testing can, in fact, take us far beyond that.'
>
> Jonathan Kohl (Kohl, 2013)

The coverage of testing can be defines as: '*The amount of testing a software using different approaches'* (Kohl, 2013). A coverage model describes how the software is tested, as different testers do the testing, the software will be tested from different perspectives which will ensure a better coverage. Jonathan Kohl quotes James Bach in his article 'Demystifying Exploratory Testing' (Kohl, 2013) on six possible models of coverage for exploratory testing;

- The structure of the application.
- The function of the application.
- The data the application uses.
- The platform the application runs on.
- The operations the applications users typically engage in.
- The impact of time on the system.

Coverage outlines can help the testers keep track of what have been tested and when (Kohl, 2013). Coverage outlines can be physical documents in the form of checklists or visual diagrams that shows the user's flow through the system. The purpose is to direct, guide and communicate what was tested and why. There is no toolbox for exploratory testing, testers develop their own toolboxes of ideas and tools out of experience (Kohl, 2013).

> 'This knowledge comes from our testing experience, as well as books, Websites, product production, subject matter experts, other team members, and conference, course, or workshop materials... The richer our information sources, the more informed our testing will be.'
>
> Jonathan Kohl (Kohl, 2013)

To guide the exploratory testing there is often some sort of guide. Different types of test guides can be found along with the exploratory testing style descriptions in appendix 10. Exploratory testing can be structured with well-defined guidelines or checklists, or it can be unstructured, 'ad hoc' or jumping around the application to find problems (Kohl, 2013). Exploratory testing can be combined with any kind of testing technique, such as performance, load, function, usability, accessibility, security, etc. (Kohl, 2013). No specific skills are necessary to perform exploratory

testing, but the most common skills identified with exploratory testers, according to Jonathan Kohl (Kohl, 2013), are the following:

- **Strategic thinking:** Pick an area of focus and adjust the testing to achieve the best coverage.
- **Idea generation:** To be able to think in different ways, and be able to adapt their testing.
- **Investigation:** Analysis, find areas of interest and explore, using tools, processes and testing techniques.
- **Communication:** Documenting the approach and explain the strategy and the problems which were found.

The output of exploratory testing is to aid the stakeholder and the team help make decisions about the software (Kohl, 2013). The output information to be communicated can be in the form of bug reports, notes, session sheets, status reports, and test coverage reports, quality criteria, blocking issues (problem areas) or effort and priorities.

### 4.5.5 Summary of the theories and methods of software testing

All programmers perform white-box testing to some extent. To most; it comes naturally to run every function as soon as it has been written. How else will the programmer know if a comma or slash is missing?

As soon as the programmers feel confident that the functions, which they have written individually are in order, grey-box and black-box testing can start. This includes testing that all functions responds as expected, when using the application, and that one function does not exclude another. But it is well known that no software is without flaws, there are limitations as to the amounts of testing being performed and also to the testers' and users' expectations as to how the system is to be used, regardless of thorough and comprehensive documentation compiled in conjunction between the users and the producers.

The benefits of exploratory testing according to Juha Itkonen and Kristian Rautiainen (Itkonen & Rautiainen, 2005), is as follows: *'...include the versatility of testing and the ability to quickly form an overall picture of system quality.'* One shortcoming of exploratory testing is managing the test coverage. It is difficult to ensure the coverage of testing without a fixed procedure for testing, and testing in all types of environments; in-door, out-door, sunshine, rain, frost and snow, at peaceful times and during life critical emergency situations. The more advanced the system and the technical devices become, often they become more sensitive to their environments and the way they are handled. Preferably; in relation to testing, our system should also be tested during the worst of situations; with both extreme rain and risk of lives and values. But this will not be a possibility for us, as we cannot induce a situation of extreme rain at will or ask emergency commanders to test our system during life critical situations.

## 4.6 Summing up on theories and methods

In this chapter we have been through theories and methods concerning e.g. databases, interfaces, web access, servers, geographic positioning systems and software testing, etc. This chapter has provided us with fundamental knowledge on which to base our own experiences. In

the next chapter we document our experiences with building an application founded on the knowledge obtained in this chapter.

From this chapter; we now know about designing interfaces for mobile devices and computers with larger screens, and the interaction between humans and computers. We have looked into possibilities of sharing data with other users, developing prototypes, system architecture and setting the databases, receiving signals from global positioning systems and software testing. In the next chapter we introduce our own work, and methods used through the progress of this project.

# 5 Technical implementation

Here in chapter 5, it is our goal is to answer the 4 remaining research questions listed in the Introduction in section 1.2 with our own experiences. The results which we gain in this chapter will be the base for solving our problem statement. Just to refresh the research questions, they are listed here:

- '*How can we establish and host a database to support updating?*'
  Section 5.1: 'Sharing a database'
- *'How can we design an appropriate user interface for emergency response work?*'
  Section 5.2: 'Our user interfaces for the field officers and the NOST'
- '*How can we develop a prototype, a mobile application based on Web standards and open source products, meeting some of the needs of the emergency rescue teams and NOST?*'
  Section 5.3: 'Our system and challenges' and section 5.4: 'Technique, programming.'
- '*How can the developed prototype tool be used as a human rescue and warning system?*'
  Section 5.5: 'Our prototype as a rescue and warning system'

In section 5.6 'Performed Software Testing', we document our testing of function and capabilities, and analyse our test results and evaluate the gains achieved by testing.

In short this chapter will guide our readers thought our work and efforts towards a practical prototype of a system for the Danish Emergency Rescue Services (DERS). We have been through a complex process of trying out theories, and the evaluation of every step of our progress.

## 5.1 Sharing a database

Here we explore the practical side of defining and importing data in PostGIS and setting up services in GeoServer to support our prototype. We will also focus on security and how to improve access speed.

### 5.1.1 PostGIS database

The main purpose of the database is to store and share data, e.g. when the emergency rescue leader defines the focal area or inserts roadblocks. The registered data in the database allows other participants like NOST users to see registrations being made from the rescue area. Furthermore queries can be made in SQL, based on the information stored.

The needed data stored in PostGIS for our prototype is shown in Table 5-1. Some of the data sets like Address Points and Road Names are large data sets which have been imported from the Ministry of Housing, Urban and Rural Affairs (MBBL), whereas the table of 'Focal Area' has been defined in the database without any features. The reason for this is; that features are created in the field by the emergency commanders while estimating the situation.

| Data name | Data type | Geometry | Purpose |
|---|---|---|---|
| Address points | WMS/WFS | point | Meeting points, query |
| Buildings | WMS/WFS | polygon | Defining building types |
| Focal area | WMS/WFS-T | polygon | Displaying the focal area |
| Terrain depressions | WFS | polygon | low area, flooding risk |
| Road blocks | WFS-T | point | Access overview |

Table 5-1: listing the data tables defined in PostGIS (MTM group 4)

Address Points and Road Names are provided by the MBBL. MBBL is the overall authority responsible for providing road names and addresses in Denmark. Data can be downloaded in several formats like CSV, WMS and WFS. The data for Address Points has been downloaded in CSV format and sorted by the municipality code, to reduce the amount of data from all of Denmark to Copenhagen and suburbs, which is our area of interest. By doing this we reduce the amount of data from 1.2 million rows to approx. 0.267 million rows or from 34.8 million data posts to 7.7 million data posts. This is still a lot of data, and it raises considerations as to whether the data for our solution can be further reduced to improve the response time. One way to achieve this; could be to delete the columns which are unnecessary to the officers in the field. But this will require automated routines every time an update is conducted. Using this method will require keeping a copy of the complete database for the NOST users; who might need access to the complete data set. Selecting the columns can be done using SQL script, e.g. by creating a SQL script selecting the wanted columns.

```
SELECT Column1_Name, Column2_Name, .. ColumnN_Name
FROM table_name;
```

The Figure 5-1 below illustrates a map overlaid with the attribute table for Address points. In this figure the area of interest is defined by blue Address points, please note the obviously errors where the Address Points are placed in water south of Greve in Øresund and by Strøby Egede. This issue raises another discussion of quality of data and the data responsibilities of the data owners for ensuring the data quality e.g. accuracy. The quality of data is further discussed in the 6th chapter; *Discussion*.

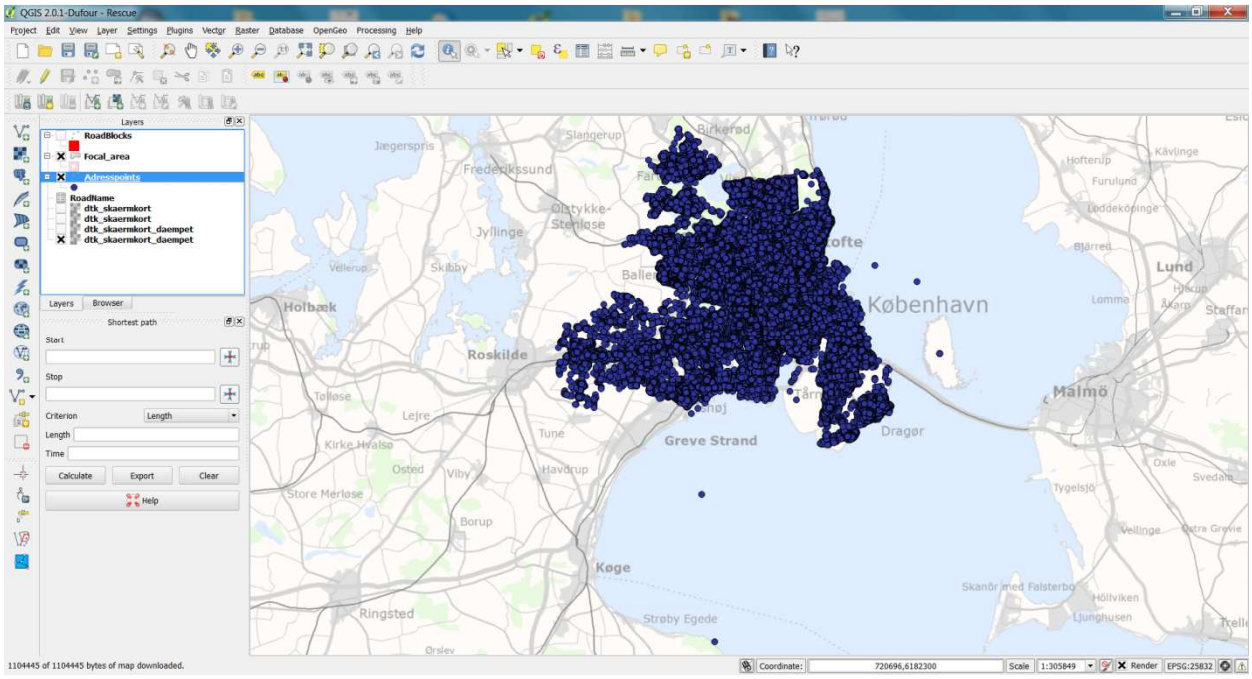



**Figure 5-1: Screen-dump created in QGIS, illustrating an area of interest overlaid by Address Points (MTM group 4).**

PostGIS has a Shapefile Import/Export Manager interface: shp2pgsql, where it is possible to load existing data; e.g. our Address Points into the database from shapefiles *.shp. In order to import data, access to our database has to be established. This is established by giving reference to; the defined database in PostGIS, the port connection accessing PostGIS, the server host, username and password. Since PostGIS is installed on the computer acting as a server, the server host is defined as localhost. The port connection setting for PostGIS is by default set to 54321 or 5432. The file wanted for import has to be selected under add file and the character encoding has to be written under options. Data will be placed by default under the public schema in PostGIS; when importing data with shp2pgsql. Schemas are like folders, and can hold tables, views, and functions. For 'Address Points' the character encoding must be Latin1 to ensure that the Danish characters like æ, ø and å are imported correctly. Latin1 is a defined ISO standard: ISO 8859. Figure 5-2 illustrates the 'PostGIS Shapefile Import/Export Manager' for shp2pgsql. It is our experience that getting the projection Spatial Reference System Identifier (SRID) from the *.proj file, is not automated, and therefore it is necessary to manually add the SRID projection as visualised in the figure. This is important because without SRID which is information about the geographic coordinate system and projection, the data is useless in the PostGIS database.

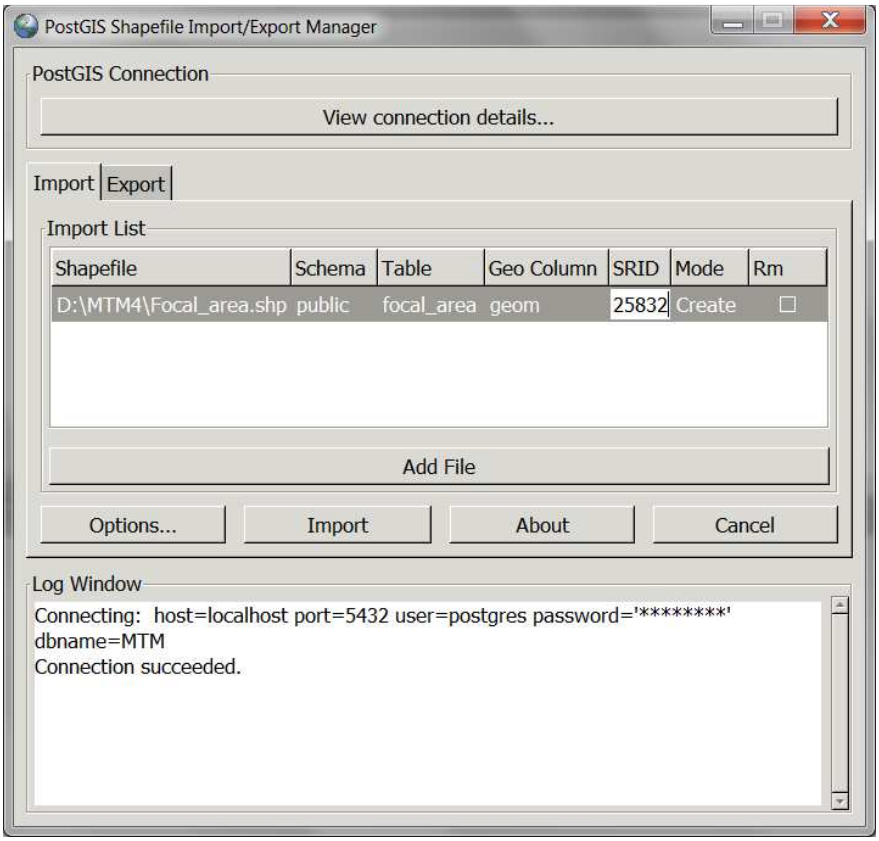



Figure 5-2: Database access pane in shp2pgsql interface (PostGIS Shapefile Import/Export Manager) (MTM group 4)

As a second option we have decided to import data into PostGIS, by using the QGIS (Quantum GIS) build-in plug-in SPIT (Shapefile to PostGIS Import Tool). It is possible, by using this function, to import multiple shapefiles at the same time into the PostGIS database. The next figure, Figure 5-3, illustrates the SPIT connection pane for importing shapefiles in QGIS. In order to import data, the same access information has to entered; as with shp2pgsql. The only difference is the option of selecting; which schema to import data into. The_geom, SRID (Spatial Reference system IDentifier) and GID (Geographical IDentifier) are set using the SPIT connection pane. The_Geom defines the geometry type of PostGIS, SRID the reference system, and GID the primary key column. Global schema defines in which schema to place the data, as every database can have several schemas. The Public schema is the default schema in PostGIS, and if no reference is given; when importing data, data will by default be placed here. The SPIT also includes Filename; naming the shapefile and locating the source file, Feature Class; identifying the geometry type, Features; count of object in the file, and DB Relation Name; is the table name in PostGIS.

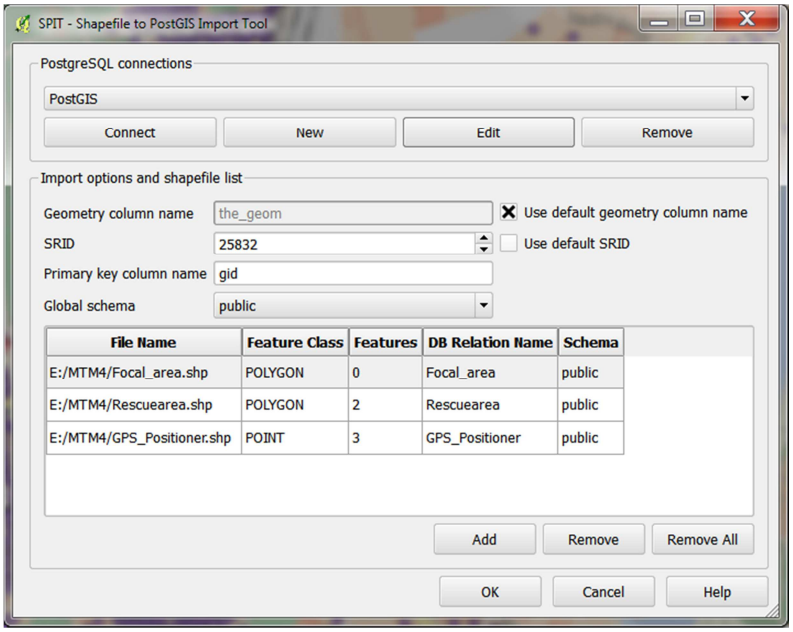



Figure 5-3: SPIT (QGIS) connection pane for shape file imports into PostGIS (MTM group 4)

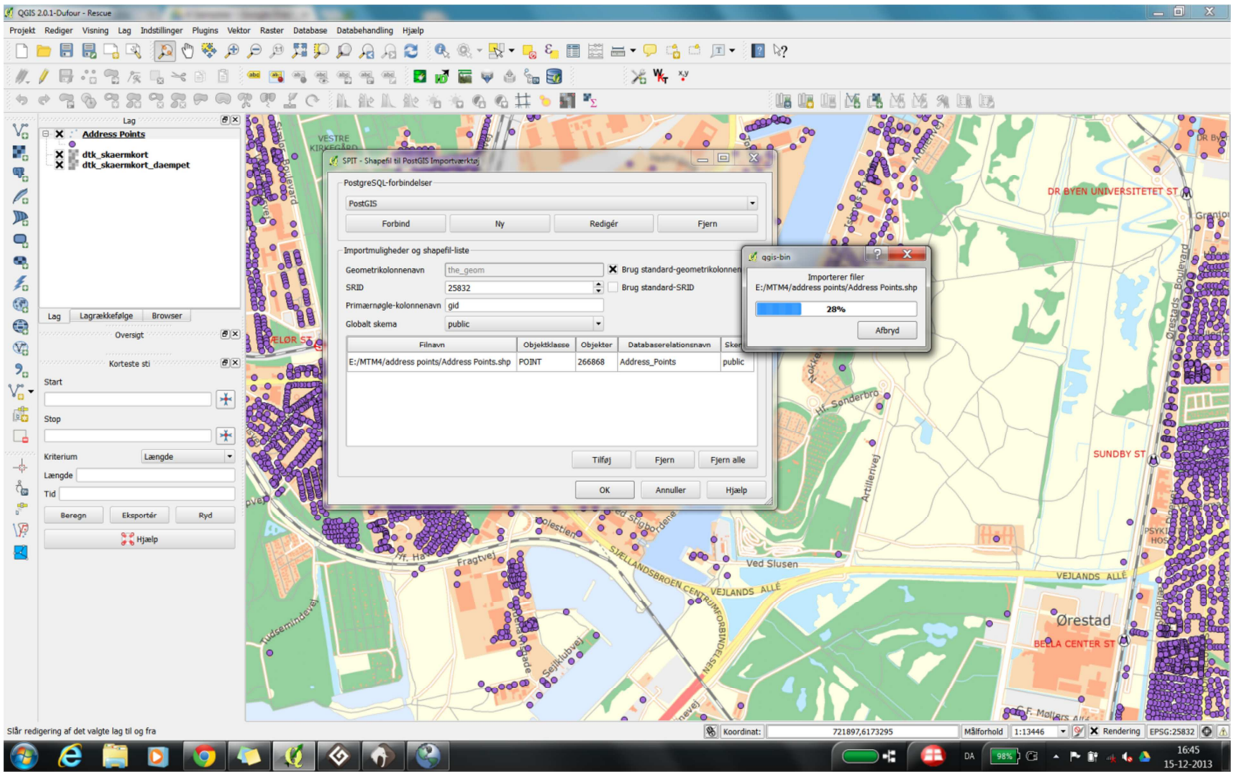



Figure 5-4: SPIT user interface and database connection pane from QGIS. Showing loading of the 'Address Point' shape layer into PostGIS (MTM group 4)

In general the import of large data sets can be troublesome regardless of the software used. Figure 5-4 is a screen dump of the loading of Address Points into PostGIS, the Address Points data set contains 266868 features or rows to be imported. To check the import has been successful we turn to pgAdmin. The pgAdmin user interface has a tree structure showing the created databases in PostGIS, and the assigned schemas and tables. Figure 5-5 illustrates the pgAdmin user interface, and the properties of the imported Address_Points table with 266868

rows. The Data view option also allows viewing data directly in the table. This information indicates whether the data import has been succeeded correctly.

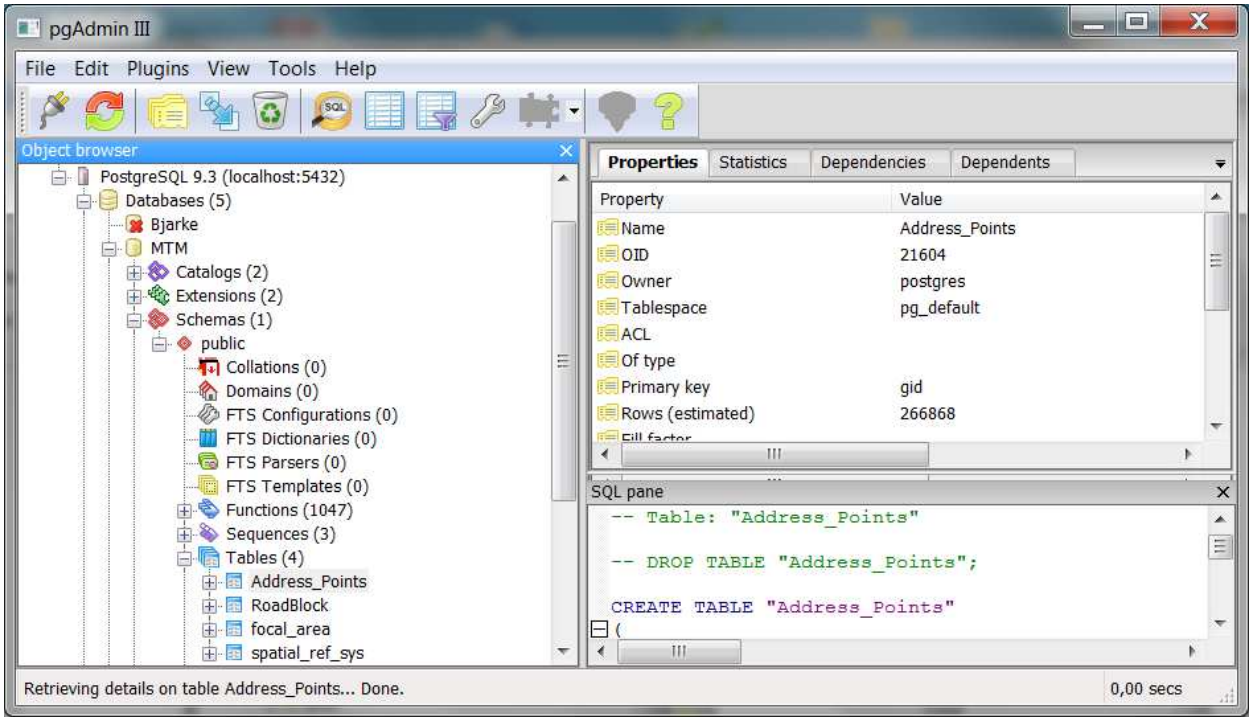



Figure 5-5: pgAdmin user interface showing the imported 'Address points' in PostGIS (MTM group 4)

We have used 2 methods for importing data sets into our PostGIS database; PostGIS's own import/export manager and QGIS. Our preference is SPIT which is a part of QGIS, as SPIT contains the possibility of importing more than one data set at the time. With pgAdmin we have checked the import of the data set, a further assurance of the success of the import is to use the GeoServer to display the data set over the internet.

### 5.1.2 GeoServer

GeoServer is a powerful open source application for displaying data over the internet, because it is compatible with many databases and can publish data in many different standards and formats. In our project we have chosen to publish our data in WMS, and WFS (WFS-T) format. As long as our project is just a prototype and we want to show how the data should be used and published, we are working with a few sets of data instead of composing comprehensive system containing enormous amounts of data. We have selected to use buildings, roads, address points and terrain depressions (see Table 5-1 in section 5.1.1). We have created two tables in our database for publishing the data as WFS-T format, one for Focal Area and one for Roadblock. Publishing data as WFS-T makes it possible to create, edit, and delete objects. During rescue operations; DERS can make and delete focal areas or roadblocks. The layers of roads and buildings are published as WFS format, giving the possible to create queries forming the requested data. Please note that this function is not active in our application.

It is essential for DERS to identify the buildings such as schools, nurseries and nursing homes with vulnerable people and help them under evacuation. At this point there is no common database or dataset for these buildings, we see this as a problem. Our data; which contains

buildings is downloaded from GST and it contains a column with the name of building types. The purpose of this column is for the municipalities to include information on the use of the buildings, the problem is that there is no legal obligation for them to do this.

To display our data as WFS on the GeoServer, firstly we created a workspace to group the data as shown in Figure 5-6.

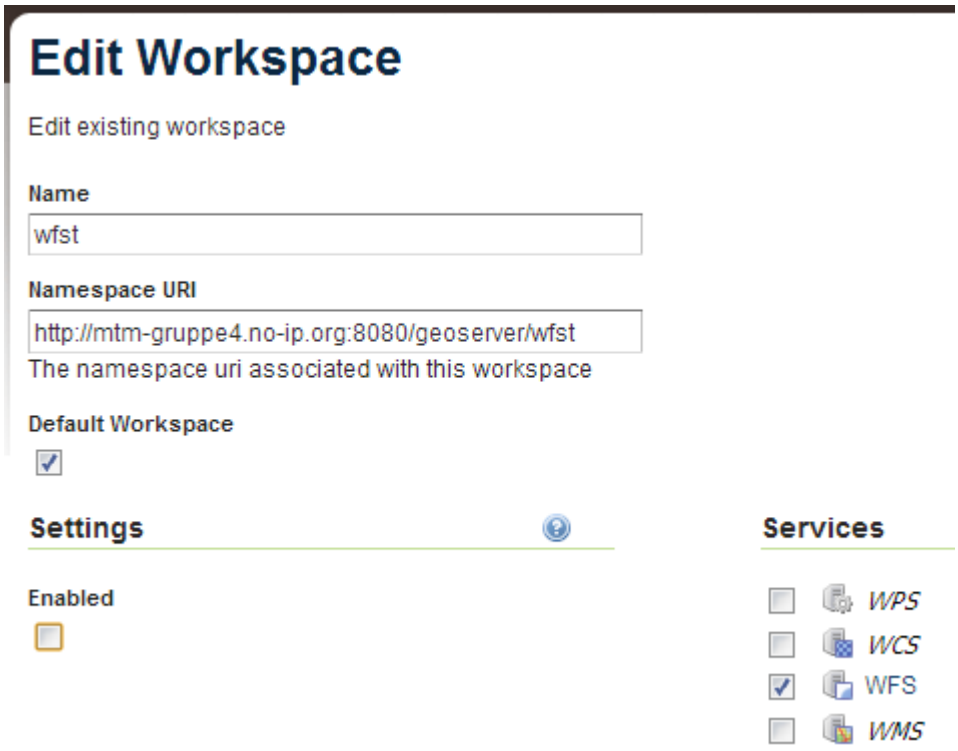



Figure 5-6: Creating a workspace in GeoServer (MTM group 4)

To create a workspace; it is necessary to name it and introduce the 'namespace URI (Uniform Resource Identifier)' which defines the web reference. URI is an identifier which stands for Uniform Resource Identifier identifies the provided workspace from the server, whereas the URL (Uniform Resource Locator) identifies the location of the displayed service.
We have made a connection between our Workspace and our data PostGIS database by adding a new Source under Workspace, referring it to the PostGIS database, and inserting the connection parameters and username and access code, please see Figure 5-7.

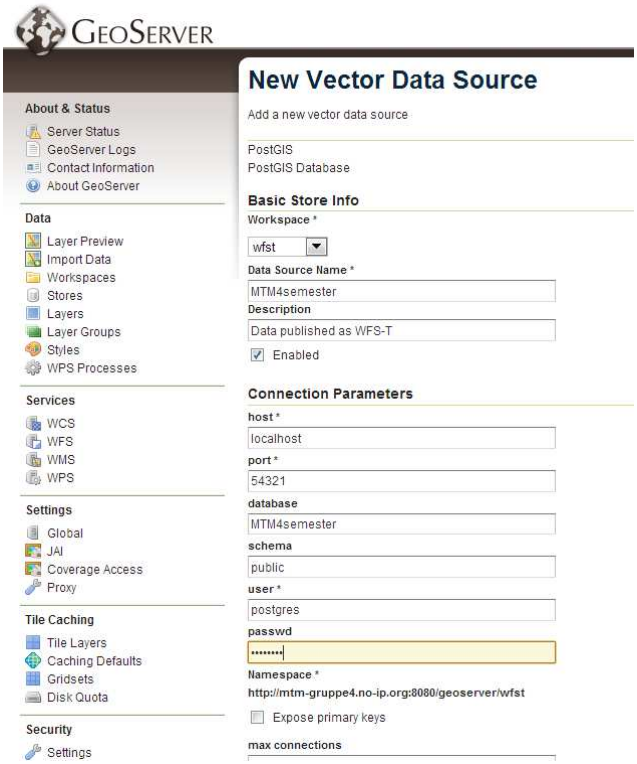



Figure 5-7: Illustrates the creation of a Store, and creating a connection to the PostGIS database (MTM group 4)

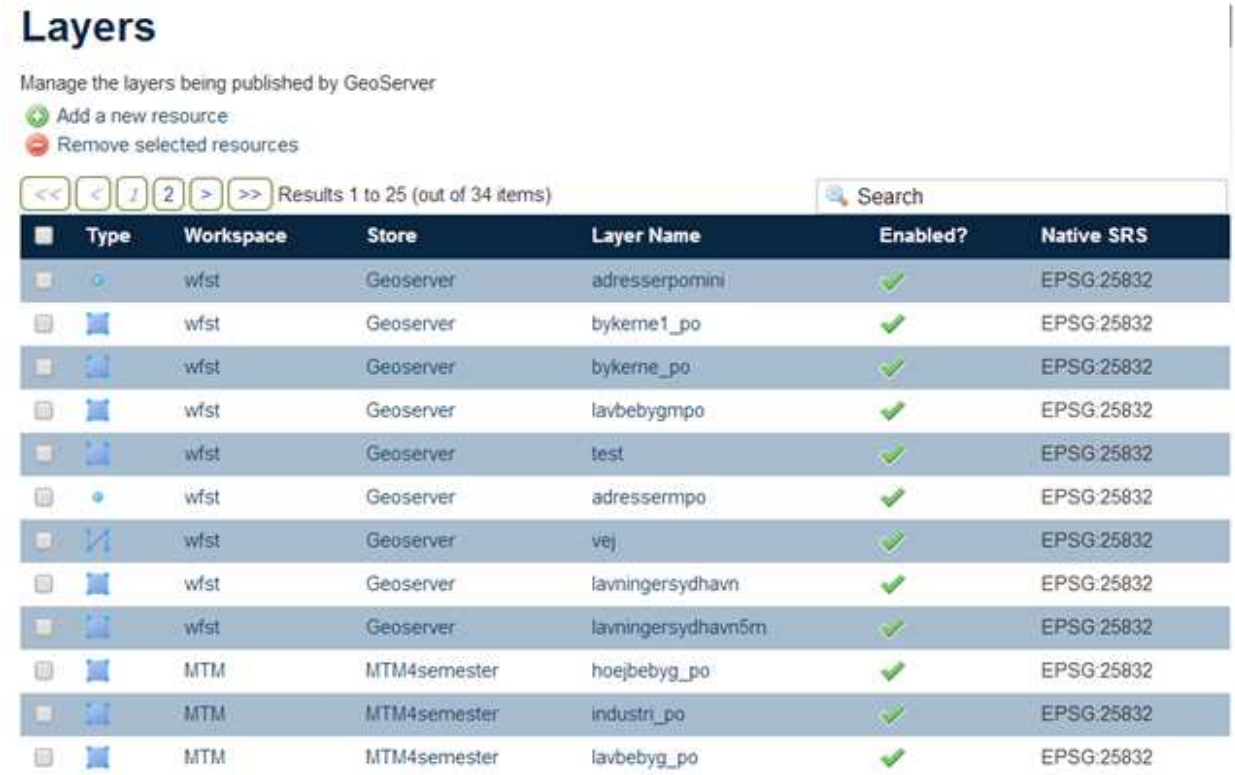
When connection is established the available layer can be chosen for publishing. Having different workspaces and stores After connecting the Workspace to our data in PostGIS database we have styled to our data. By using different and suitable colours, shapes, opacity and thickness, it is possible to open more data layers and get different information at the same time in our application. By editing in a XML file we can define the desired style for output in GeoServer, Figure 5-8.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<StyledLayerDescriptor version="1.0.0" xmlns="http://www.opengis.net/sld" xmlns:ogc="http://www.opengis.net
  xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/sld http://schemas.opengis.net/sld/1.0.0/StyledLayerDescriptor
  <NamedLayer>
    <Name>Default Polygon</Name>
    <UserStyle>
      <Title>Default polygon style</Title>
      <Abstract>A sample style that just draws out a solid gray interior with a black 1px outline</Abstract
      <FeatureTypeStyle>
        <Rule>
          <Title>Polygon</Title>
          <PolygonSymbolizer>
            <Fill>
              <CssParameter name="fill">#996633</CssParameter>
              <CssParameter name="fill-opacity">0.3</CssParameter>
            </Fill>
            <Stroke>
              <CssParameter name="stroke">#000000</CssParameter>
              <CssParameter name="stroke-width">0.1</CssParameter>
            </Stroke>
          </PolygonSymbolizer>
        </Rule>

      </FeatureTypeStyle>
    </UserStyle>
  </NamedLayer>
</StyledLayerDescriptor>
```

Figure 5-8: A screen-dump of how to edit the style in XML file in GeoServer (MTM group 4)

The layer menu displays our published layers available for web applications. It gives an overview of layers, stores, workspaces and type (Figure 5-9).



Layers

Manage the layers being published by GeoServer

Add a new resource

Remove selected resources

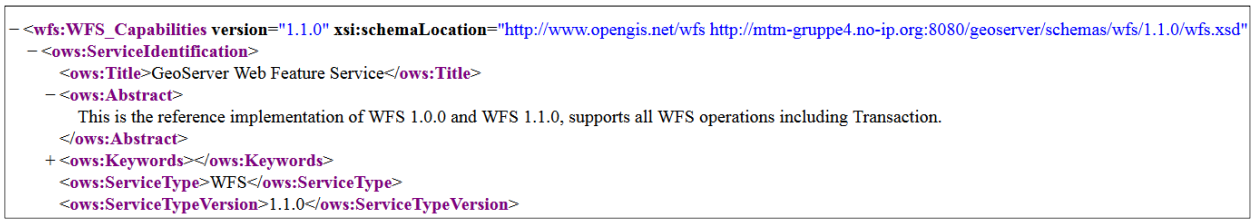
<< < 1 2 > >> Results 1 to 25 (out of 34 items) Search

| Type | Workspace | Store | Layer Name | Enabled? | Native SRS |
|---|---|---|---|---|---|
| | wfst | Geoserver | adresserpomini | ✓ | EPSG:25832 |
| | wfst | Geoserver | bykerne1_po | ✓ | EPSG:25832 |
| | wfst | Geoserver | bykerne_po | ✓ | EPSG:25832 |
| | wfst | Geoserver | lavbebygmpo | ✓ | EPSG:25832 |
| | wfst | Geoserver | test | ✓ | EPSG:25832 |
| | wfst | Geoserver | adressermpo | ✓ | EPSG:25832 |
| | wfst | Geoserver | vej | ✓ | EPSG:25832 |
| | wfst | Geoserver | lavningersydhavn | ✓ | EPSG:25832 |
| | wfst | Geoserver | lavningersydhavn5m | ✓ | EPSG:25832 |
| | MTM | MTM4semester | hoejbebyg_po | ✓ | EPSG:25832 |
| | MTM | MTM4semester | industri_po | ✓ | EPSG:25832 |
| | MTM | MTM4semester | lavbebyg_po | ✓ | EPSG:25832 |

Figure 5-9: Layer menu (MTM group 4)

To check whether our services are displayed over the internet, we send a request by the internet to our server at: mtm-gruppe4.no-ip.org. As mentioned in section '4.3.2 Input data', we will send a GetCapabilities request to our WFS service.[26]

The full GetCapability request can be seen in appendix 11. From the request our service is displayed and the identification set as 'GeoServer Web Feature Service' and service type is 'WFS', Figure 5-10.



```
− <wfs:WFS_Capabilities version="1.1.0" xsi:schemaLocation="http://www.opengis.net/wfs http://mtm-gruppe4.no-ip.org:8080/geoserver/schemas/wfs/1.1.0/wfs.xsd"
  − <ows:ServiceIdentification>
      <ows:Title>GeoServer Web Feature Service</ows:Title>
    − <ows:Abstract>
        This is the reference implementation of WFS 1.0.0 and WFS 1.1.0, supports all WFS operations including Transaction.
      </ows:Abstract>
    + <ows:Keywords></ows:Keywords>
      <ows:ServiceType>WFS</ows:ServiceType>
      <ows:ServiceTypeVersion>1.1.0</ows:ServiceTypeVersion>
```

Figure 5-10: Service identification and service type (MTM group 4)

Furthermore the supported output format is shown under 'Parameter' and 'Value'. The 'OperationsMetadata; shows that the WFS server supports operations like GetGmlObject, LockFeature, GetFeature WithLock and Transaction, Figure 5-11.

[26] http://mtm-gruppe4.no-ip.org:8080/geoserver/ows?service=wfs&version=1.1.0&request=GetCapabilities

```
− <ows:Parameter name="outputFormat">
    <ows:Value>text/xml; subtype=gml/3.1.1</ows:Value>
    <ows:Value>GML2</ows:Value>
    <ows:Value>GML2-GZIP</ows:Value>
    <ows:Value>SHAPE-ZIP</ows:Value>
    <ows:Value>application/gml+xml; version=3.2</ows:Value>
    <ows:Value>application/json</ows:Value>
    <ows:Value>csv</ows:Value>
    <ows:Value>gml3</ows:Value>
    <ows:Value>gml32</ows:Value>
    <ows:Value>json</ows:Value>
    <ows:Value>text/xml; subtype=gml/2.1.2</ows:Value>
    <ows:Value>text/xml; subtype=gml/3.2</ows:Value>
  </ows:Parameter>
 + <ows:Constraint name="LocalTraverseXLinkScope"></ows:Constraint>
 </ows:Operation>
+ <ows:Operation name="GetGmlObject"></ows:Operation>
+ <ows:Operation name="LockFeature"></ows:Operation>
+ <ows:Operation name="GetFeatureWithLock"></ows:Operation>
+ <ows:Operation name="Transaction"></ows:Operation>
</ows:OperationsMetadata>
```


Figure 5-11: WFS Output format (MTM group 4)

FeatureType are shown in the form namespace:featuretype. The default projection of the feature type is also listed, along with the bounding box for the data in the stated projection. Figure 5-12 shows MTM:hoejbebyg_po with keyword projection and boundingbox.


```
− <FeatureType>
    <Name>MTM:hoejbebyg_po</Name>
    <Title>Hoej bebyggele i project område</Title>
    <Abstract/>
  − <ows:Keywords>
      <ows:Keyword>features</ows:Keyword>
      <ows:Keyword>hoejbebyg_po</ows:Keyword>
      <ows:Keyword>high buildings</ows:Keyword>
    </ows:Keywords>
    <DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
  − <ows:WGS84BoundingBox>
      <ows:LowerCorner>12.252697461032465 55.55303406957044</ows:LowerCorner>
      <ows:UpperCorner>12.695031134983887 55.87068559083786</ows:UpperCorner>
    </ows:WGS84BoundingBox>
```


Figure 5-12: MTM:hoejbebyg_po (MTM group 4)

Now we want see if it's possible to get data from the featuretype 'lavningersydhavn', by sending a request on returning the geometry geom.[27]

Figure 5-13 shows the return of the request with geometry values from the feature 'lavningersydhavn', and from that we can conclude that we have access to the PostGIS database through GeoServer.

[27] http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs?service=wfs&version=2.0.0&request=GetPropertyValue&typeNames=wfst:lavningersydhavn&valueReference=geom

```
− <wfs:ValueCollection xsi:schemaLocation="http://www.opengis.net/wfs/2.0 http://mtm-gruppe4.no-ip.org:8080/geoserver/schemas/wfs/2.0/wfs.xsd">
  − <wfs:member>
    − <wfst:geom>
      − <gml:MultiSurface srsDimension="2" srsName="urn:ogc:def:crs:EPSG::25832">
        − <gml:surfaceMember>
          − <gml:Polygon srsDimension="2">
            − <gml:exterior>
              − <gml:LinearRing>
                − <gml:posList>
                    723724.7999999999 6173596.8 723744.0 6173596.8 723744.0 6173577.6 723734.3999999999 6173577.6 723715.2 6173577.6 723715.2 6173567.999999999 723705.6 6173567.999999999 723686.399999999
                    6173567.999999999 723686.3999999999 6173558.399999999 723676.7999999999 6173558.399999999 723667.2 6173558.399999999 723667.2 6173529.6 723628.7999999999 6173529.6 723628.799999999
                    6173558.399999999 723638.3999999999 6173558.399999999 723648.0 6173558.399999999 723648.0 6173567.999999999 723657.6 6173567.999999999 723667.2 6173567.999999999 723667.2 6173577.6
                    723696.0 6173577.6 723705.6 6173577.6 723705.6 6173587.199999999 723715.2 6173587.199999999 723724.7999999999 6173587.199999999 723724.7999999999 6173596.8
                </gml:posList>
              </gml:LinearRing>
            </gml:exterior>
          </gml:Polygon>
        </gml:surfaceMember>
      </gml:MultiSurface>
    </wfst:geom>
  </wfs:member>
  − <wfs:member>
    − <wfst:geom>
      − <gml:MultiSurface srsDimension="2" srsName="urn:ogc:def:crs:EPSG::25832">
        − <gml:surfaceMember>
          − <gml:Polygon srsDimension="2">
            − <gml:exterior>
              − <gml:LinearRing>
                − <gml:posList>
                    723744.0 6173529.600000001 723772.8 6173529.600000001 723772.8 6173520.0 723744.0 6173520.0 723744.0 6173529.600000001
                </gml:posList>
              </gml:LinearRing>
            </gml:exterior>
          </gml:Polygon>
        </gml:surfaceMember>
```


Figure 5-13: Get geometry values from 'lavningersydhavn'

We have explained our connection to the GeoServer with the PostGIS database and how to create a new workspace, new store, style our layers, and publish our data. We have sent a request to get information and data from our web service.

In the next chapter we write about the considerations; which we have taken regarding the security and maintenance of our system.

## 5.2 Our user interfaces for the field officers and the NOST users

To fulfil the needs of the emergency commanders operating in the field and the needs of the NOST officers at the Command centre, 2 user interfaces are needed. Due to time limitations for developing the user interfaces, this section will mainly focus on the user interface for emergency commanders, who we plan to equip with tablets for using in the field.

We document the progression of our user interface development for the emergency commanders. From the basic paper mock-ups to the more advanced digital prototype for emergency commanders in their operations. Many ideas have been put into the design of these mock-ups, as we have shown and discussed through this entire project. In appendix 12 can be seen a long line of mock-up generations.

### 5.2.1 Prototype A

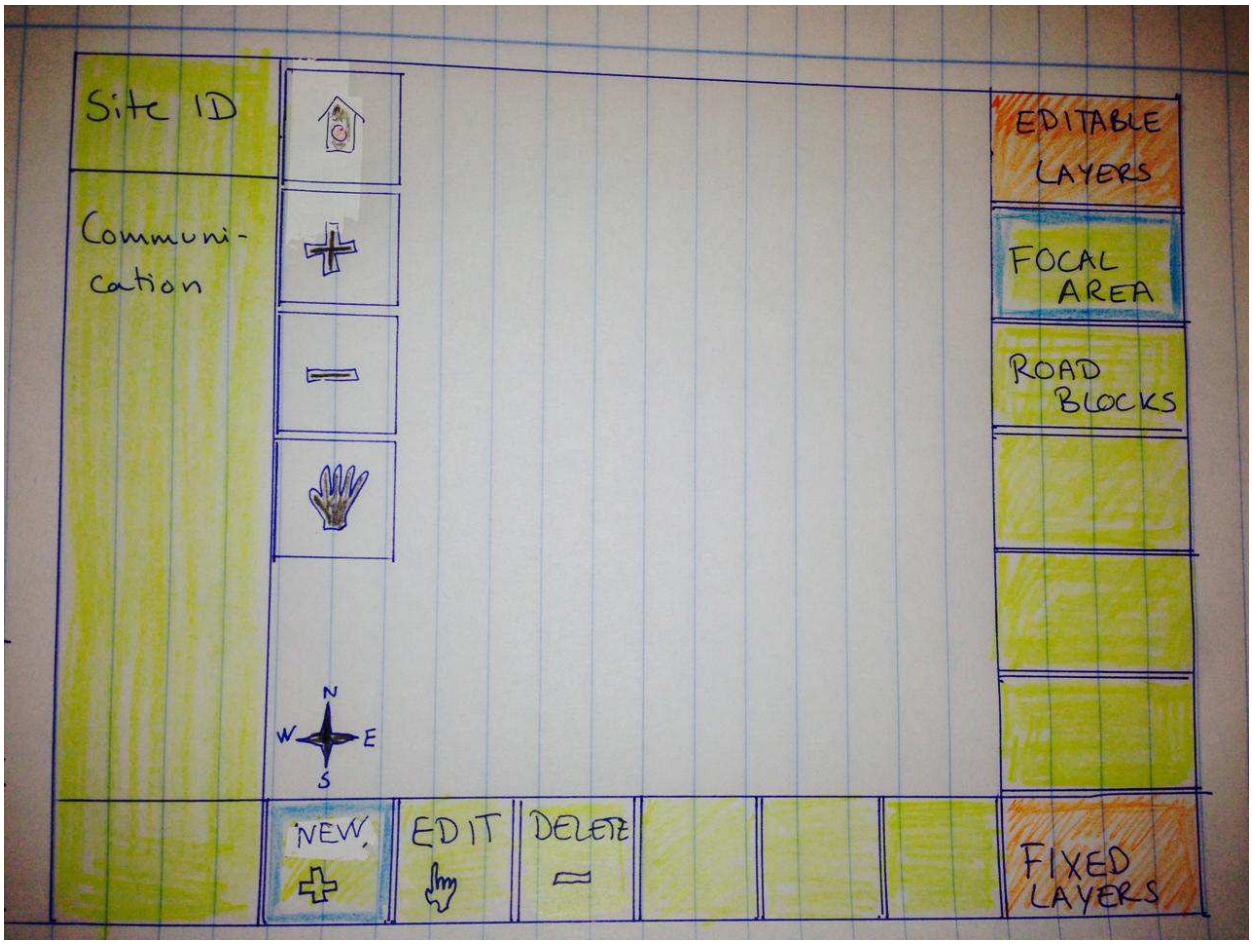



Figure 5-14: Prototype A (MTM group 4)

Figure 5-14 is a photo of our mock-up called Prototype A. The user interface is set to match use in the field. In the left hand side of the screen there is room for communicational functions. In the middle of the screen is the map view. The uppermost tool in the left side of the map view is the 'Home' button (drawing of a house with a red dot on the front), which will zoom/pan the view to the users current location. Just below the 'Home' button is the '+' button, this will zoom-in in the map. Next is the '-' button which zooms out. Below the '-' button is the 'Hand' button, which should make it possible to pan the map. In the lower right corner of the map view is the compass rose. To implement a compass rose in the map makes it necessary to start a discussion on whether the compass rose should always point to the magnetic North or the north of the map view. This discussion will be taken up later in chapter 6; *the discussion*.

Outside the map view in the right hand side of the screen; the layers are listed. Primarily the editable layers are listed, e.g. 'Focal area' and 'Road blocks'. In case it is necessary to identify or alter the order of the fixed layers, using the 'Fixed layers' button will give a list of the fixed layers. Below the map view in the bottom of the screen are the buttons with the functions to alter the editable layers. Starting from the left hand side is the 'New' button, which will aid the user to add objects to the editable layers. Next is the 'Edit' button, which is to be used when an object needs to be altered. The last button is 'Delete', to be used when an object must be deleted.

The expected users of this prototype are the emergency commanders, who operate in the field during emergency operations. The basic use of this prototype is for the emergency commanders to gain information and to share their information with other users. We intend the use of our application to be both intuitive and logical.

For example: logical thinking of a user:

- 'I want to alter the 'Focal layer', therefore I select the 'Focal layer''
- 'The alteration is a new 'Focal layer' polygon, therefore I select 'New'', I draw a polygon

The Basic map to be inserted among the ‘Fixed’ layers is to be as informative as possible and still not so informative that it appears confusing to the users. To start with we introduce a topographic map and an orthophoto as background layers. The topographic map and the orthophoto are listed under the 'Fixed layers', together with the address points marking the residents of the local areas.

### 5.2.2 Prototype B

The mobile solution is designed for tablets with 10-inch screen for field use. Focus was on the needs of the user when he/she is out in the field. Touch-based user interface has become the standard for smartphones and tablets. Being able to manipulate information on the screen directly with your fingers, give the users a stronger sense of being in control of the system rather than system dictates the workflow. Another aspect of the direct interaction is that, it can make the product available for a wide range users combined with a good visual and functional interface and enhance the overall user experience. However, there are a number of disadvantages with a UI designed for mouse and keyboard interaction, they cannot be applied directly, but must first be redesigned to work with touch. Nevertheless, combined with a good visual and functional interface, touch screen can improve the overall user experience. The challenge was how to design a UI with touch and be aware of that hand and fingers can block the screen. Tablet prototype contains less function compared to desktop version, but larger buttons. Applications screenshot is shown down below in the Figure 5-15:

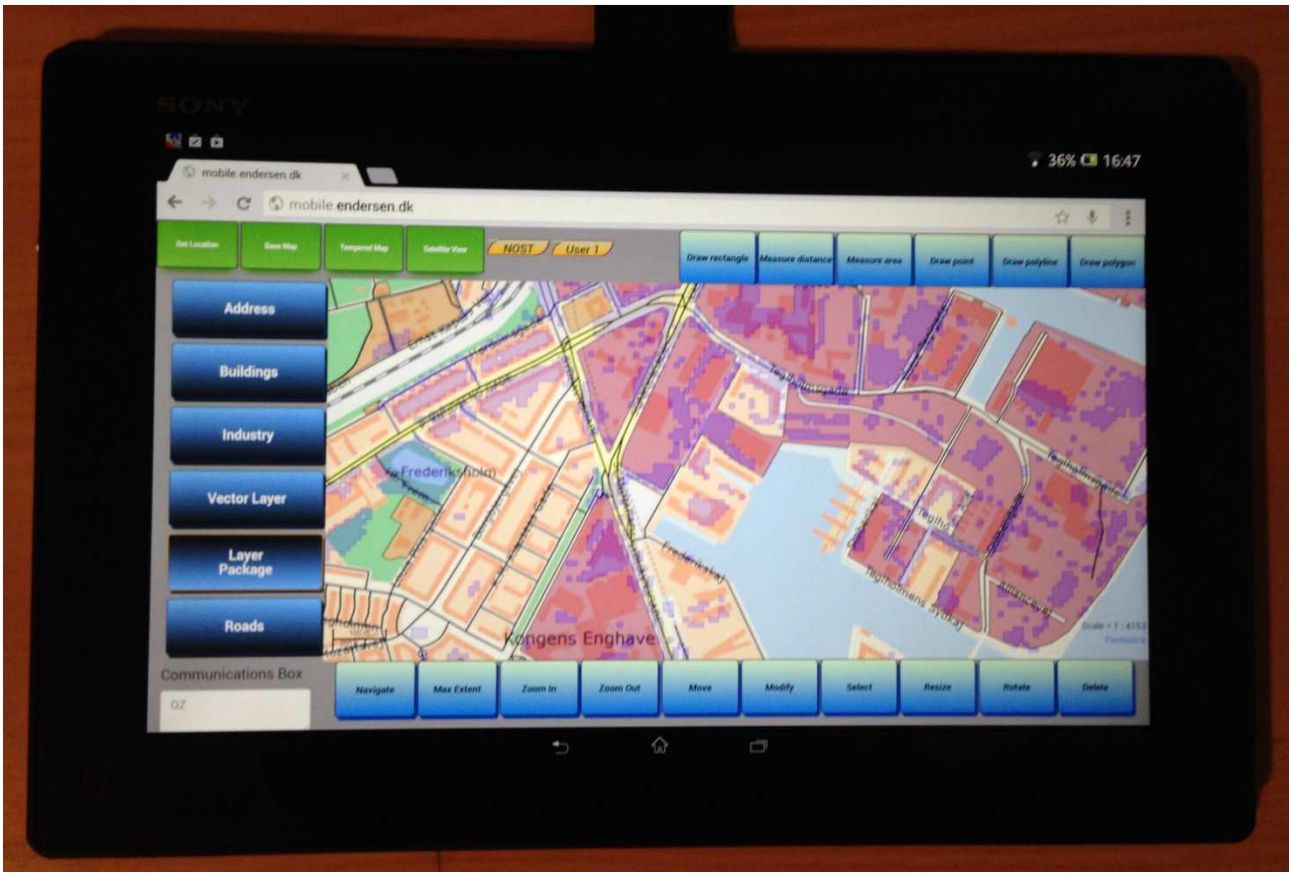



Figure 5-15: Is a photo of our tablet pc mock-up called Prototype B (MTM group 4)

### User interface flow

In the use flow list below; the flow of clicks and drags to be used when operating our application can be visualized. At first the software is activated by clicking an icon on the 'starting' screen of desktop. Secondly, before the software is completely opened, it will be necessary to choose between on a list of incidents which are available at this point in time. Of course there should not be a long list for the personnel in the field to go through, but all the simultaneous incidents should be listed (the list is only edited by personnel in the control room). After selecting the incident and the software is open, the user must determine which function is to use; create a polygon, edit a polygon, rearrange map view, insert road block or edit road block. As default the data in the map view is centred on the users location (geolocation, gps), it is also possible to zoom in, zoom out, pan and return to the users location.

### Use flow list:

**Primary actions:**

- Open internet application http://mobile.endersen.dk/
  - The application opened onscreen

**Flow 1:** *I want to select between the layers visible in my map view*

- By tabbing the layer buttons in the left side of the screen, I can add and remove the individual layer from my map view

**Flow 2:** *I want to be informed of the situation at the other users*

- **Flow 2a**
    - Tab the button 'User X'
    - The current map view of user X appears in my map view
- **Flow 2b**
    - Tab the button 'NOST'
    - The current map view of the user at NOST appears on my screen

**Flow 3:** *I want to navigate in the map view*

- **Flow 3a**
    - Tab the button 'Zoom In'
        - The map view zooms in
- **Flow 3b**
    - Tab the button 'Zoom Out'
        - The map view zooms out
- **Flow 3c**
    - Tab the button 'Max Extent'
        - The map view zoom to the extent where all feature of interest are included
- **Flow 3d:**
    - Tab the button 'Navigate'
        - Now it is possible to pan around in the map

**Flow 4:** *I want to change my background image*

- Tab the button 'Base Map', 'Tempered Map' or 'Satellite View'
    - The background image changes

**Flow 5:** *I want to get my location*

- Tab the button 'Get Location'
    - The map view identifies the users location and places a marker

**Flow 6:***I want to create a new feature*

- **Flow 6a:** *I want to draw a polyline*
    - Tab ' Draw polyline'
        - Draw polyline by tabbing the screen
            - End polyline by double tabbing on the screen
- **Flow 6b:** *I want to place a point*
    - Tab 'Draw point'
        - Tab the screen to place point
- **Flow 6c:** *I want to draw a polygon*
    - Tab 'Draw polygon'
        - Draw polygon by tabbing on the screen
            - End polygon by double tabbing on the screen

**Flow 7:** *I want to change a feature*

- **Flow 7a:** *I want to modify the shape of a feature*
    - Tab the button 'Modify'
        - Tab the feature, the feature is highlighted
            - Touch vertices and pull to the desired location and release
                - Tab the feature to de-select the polygon
- **Flow 7b:** *I want to move a feature*
    - Tab the button 'Move'
        - Tab inside the feature, the feature is highlighted

- Touch the circle marker and drag the feature to the desired location
  - Tab feature (not the circle marker) to de-select

- **Flow 7c:** *I want to resize a feature*
  - Tab the button 'Resize'
    - Touch the feature to be resized
      - Pull the marker to resize to the desired size
        - Tab the feature to end resizing
- **Flow 7d:** *I want to rotate a feature*
  - Tab the button 'Rotate'
    - Touch the feature to be rotated
      - Pull the marker to rotate the feature
        - Tab inside the feature to end rotation

**Flow 8:** *I want to delete a feature*
- Tab the button 'Delete'
  - Touch the feature to be deleted
    - Popup question appears on the screen
      - Tab 'OK'
        - The feature disappears from the screen
      - Tab 'Cancel'
        - The feature is still visible on screen

**Flow 9:** *I want to measure distance*
- Tab the button ' Measure distance'
  - Tab in the map at the starting point of the measurement
    - Tab to mark the ending point or turning point of the measurement
      - The measured distance is noted in the top right corner
        - End measuring by double tabbing in the map view

**Flow 10:** *I want to measure area*
- Tab the button 'Measure area'
  - Tab in the map at the starting point of the measurement
    - Tab the vertices of the measuring polygon
      - End the measuring polygon by double tabbing in the map view
        - The measured area is noted in the top right corner of the screen

### 5.2.3 Prototype C

Figure 5-16 illustrates our prototype C, which contains many functions. This digital prototype contains many extra buttons and functions compared to prototype A. Some of the additional buttons are the 'user' buttons, offering the user to see the map view of other users, this could be a very useful aid in regards to sharing the common image of the situation, and unfortunately this is not a working function. Furthermore the user interface of prototype C allows the user to select between different map types and also to gain information from the map view using the 'i' button.

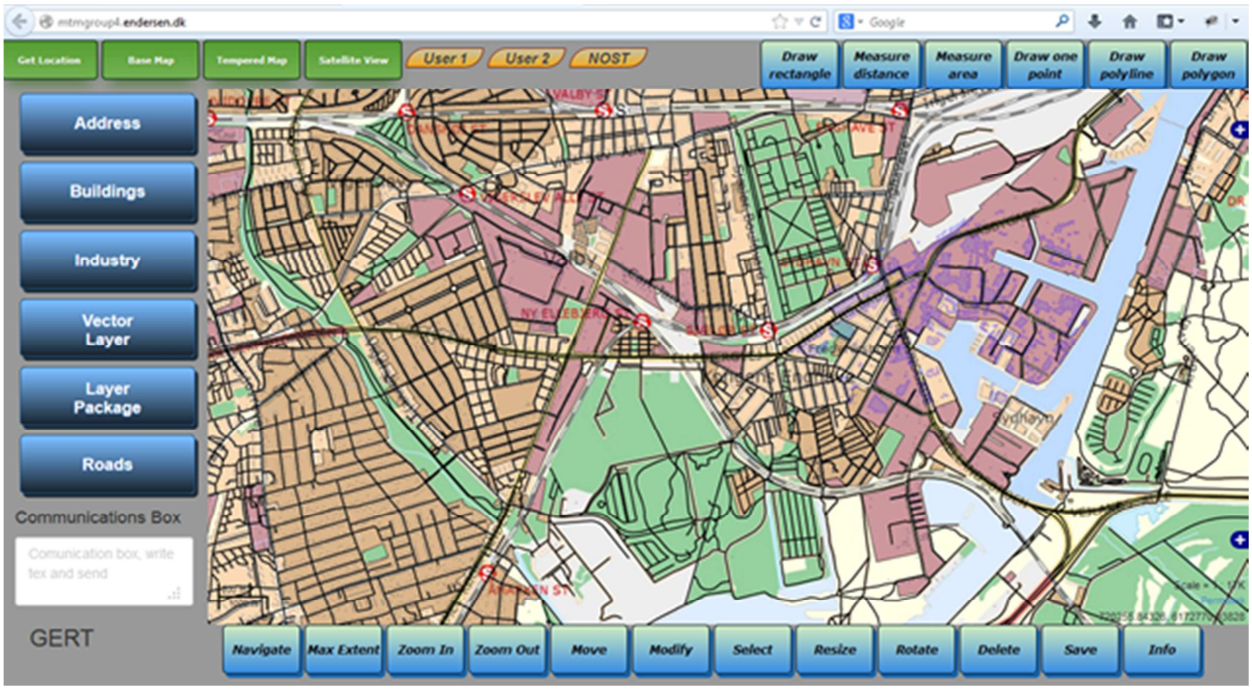



Figure 5-16: Prototype C (MTM group 4)

## 5.3 Our System and Challenges

We have selected PostGIS database for keeping and maintaining of our geodata for this project. We have installed OpenGeo Suite which is an open source package that includes PostGIS, GeoServer, GeoWebCache, GeoExplorer and ClientSDK ( Client Software Development Kit).

For safety reasons we have installed OpenGeo Suite on two different computers, therefore it will also be necessary to uphold and maintain our data on both computers. After the installation of OpenGeo Suite on our computers, it took some time for us to open the necessary ports on our internet router, so that all 4 of us could connect, gain access to the server, and the databases. The initial problem was that the internet providers had blocked some ports on the router; which we needed to be open in regards to our project. There appeared to be two possible solutions to this problem:

A. Ask the internet provider to send a new router; which allows access to configure the necessary port forwarding from LAN to WAN (Wide Area Network) site, so that the server can be reached over the internet.
B. Ask the internet provider to set the router in Bridge Mode and buy an additional router to be used alongside the first router (when a router is in Bridge mode it will just receive the public IP address and Domain Name System (DNS), from the internet supplier) with the necessary port forwarding and firewall settings is to be done by the second router.

To configure port forwarding on the router we have to log into the router's web interface (which is the same as the LAN (Local Area Network) Side gateway address, in our case 192.168.1.1.) Furthermore we had to change the port of the routers web interface, as the router used port 8080 for the internet service provider to gain access to our configuration, we changed the port to 19999, as we needed port 8080 for the GeoServer web interface.

It is important; that the windows firewall or antivirus firewall doesn't block the access to our database and server, in order to provide a service. It is possible to use websites like the gvc.com webpage to directly see what service ports are open from our own server. The ports should be open in the firewall settings as our programming model of the web applications revolves around the application server. The application's server architecture offloads most processing to the server, which hosts the application and defines the view of the web browser. Clients using a web application, such as pressing a button, activate HTML transmission to the server, ensuring that the events corresponds the user's actions. This is followed by a request from the server; in the form of a new profile image and display the form of PNG images. This is so-called thin client architecture and the advantages of this approach, according to Pinde Fu and Jiulin Sun (Fu, Sun 2011), are the following:

- *The user doesn't need to install any software other than a web browser, not even a plug-in.*
- *Since most of the heavy-lifting processes are done by the server, the client doesn't need a powerful computer, so these types of applications can work well even on low-end computers.*

*The main disadvantages are the following:*

- *Pressure on the GIS server: all GIS operations are routed to the server and processed by the server.*
- *Limited user interaction: the user interface is usually built with plain HTML and limited JavaScript, so the user interaction is not particularly smooth or compelling.*

(Fu, Sun 2011).

In contrast to a thin client server architecture there is also thick client server architecture or networks that typically provides rich functions independent of the central server. The thick client still requires at least periodic connection to a network or central server, but is often able to perform many functions without a network or server connection. A typical advantage of this architecture is increased flexibility, offline working possibilities, and lower server requirements. But limitations of this architecture are posed by the internet bandwidth and the client's computer power. Typically, it is not feasible to transfer gigabytes of data over the internet and having users perform sophisticated GIS operations. (Fu, Sun 2010).

### 5.3.1 User Access to our Data

In this section we explore the data and metadata connected to our project. We examine the quality of data, their origin, discuss how often the databases are to be updated to be kept up to date, and how can we be sure to always have the latest and most current data. Another issue is the possibility of implementing data which is not georeferenced, and how to geocode the data.

In a situation of an emergency rescue operation, it is essential that all actors involved, are always updated with the latest information. We attempt to expand our system to have the capability of sharing information between the involved parties. This can be only done by an internet connection.

There are also some questions which should be answered and considered throughout the implementation of our system, firstly: Are we are too dependent on the internet? The risks of relying too much on the internet are as follows:

- What will happen in a remote area if the internet connection is troublesome, slow or fluctuating?
- What will happen if we cannot send or receive data by the internet? What is the alternative?

Some data such as e.g. orthophotos, which we know are not to be updated frequently, can be preinstalled on the mobile device, in a low resolution, together with address points. The same procedure can be used for e.g. roads and road names from our database. With a full internet connection; the users will be able to receive data by a WMS, WFS and WFS-T from our databases, which have been discussed in the section 5.1.2, and will be discussed further in section 5.4.

Having local access to the most essential data in case of, for example, a power shortage in a local community, will aid the emergency commanders and their crews to continue their rescue operations. Having access to roads and road names will ensure the team an advantage in unfamiliar areas, to find their way around and to pinpoint their location and the location of others during radio communication with the SINE radio communications network. The advantages to be gained by having a copy of the address points on the users' tablets can be of great importance in cases of local power shortages paralyzing the telecommunication towers in the area of the impending emergency. Saving lives is of great importance to the DERS. Securing the access to address points pinpointing the location of people's homes will aid the rescue personnel to fulfil their tasks and secure evacuation.

### 5.3.2 Equipment

Based on our findings documented in our reports from the 2nd and 3rd semester, we have settled on a dual solution, including both a solution for a tablet and stationary pc solution for the NOST.

A. The tablet application is aimed for the emergency commanders, who are directly involved in the rescue operation in the field.
B. The stationary pc solution for the NOST team, sitting at the office and coordinating the situation.

The tablet should comply with the following specifications:

- A screen size of minimum 9.5 inches.
- Minimum 32 GB (Giga Byte) or 64 GB storage and 1 GB RAM (Random Access Memory).
- The battery must be able to last at least 10 hours battery of use.
- The processor must be strong, minimum 1.4 GHz.
- GPS included.
- Robust, preferably water and heat resistant, and shock absorbent.

The NOST team sitting in the command centre should have premium computers, the best possible at the present time, while the officers in the field need a solution which is hand held, fast, durable and trustworthy in situations of need.

A tablet for the emergency commanders should be small enough to be held with one hand while it should also be large enough to contain a map view and buttons for navigation, layers and functions. The minimum amount of storage is dependent on the amount of data, which we wish of download directly to the tablet, keeping the data to a minimum (orthophotos, roads, road names and address points) in low resolution; it should be possible to limit the amount to less than 32 GB. But to increase the value of the data the capacity of the device should maybe be raised to 64 GB or maybe the data should be limited to specific areas. In case of limiting the data, there will always be the risk of not having sufficient data to cover the area in question.

Regarding the duration of the battery we set the minimum boundary at 10 hours, this should ensure the operation at the most critical time, in regards to a flood resulting from extreme rain showers. Usually the Danish Meteorological Institute (DMI) warns the DERS prior to an incident of extreme rain, in which case the field teams can prepare and plan the operation in the area before the incident occurs. Warnings are posted at least 24 hours before the expected weather phenomenon occurs.[28] For this project we focus on the operation in connection with events of extreme rain; which is events of precipitation exceeding 15 mm in 30 minutes in local areas (also called storm surge).[29] 10 hours of battery time is to cover approximately 4-5 hours of preparation in the field, 30 minutes of rain showers, and the remaining time to cleaning up after the incident. A battery with longer duration would of course be preferred in these types of situations and others. The tablet should have GPS implemented to ensure the geographic referencing and geolocating. Most tablets today have processors of 1.4 GHz and 2 GHz, some with dual core and some with quad core. Of course the best and the fastest are to be preferred especially in situations of emergency and threats to human lives. But with our amount of data and only focusing on events of extreme rain 1.4 GHz and dual core should be able to cover our needs.

To demand; that the tablet must be robust and able to withstand water and heat (to some extent) is of great importance in regards to the types of operations the emergency officers find themselves in. For example the emergency commander might have to leave his car to join the team members while it is still raining to inspect a fire; caused by a short circuit due to heavy rain falls.

## 5.4 Technique and programming

Previously we have examined GeoConference (GIS system used by the police in Denmark), and GIS systems used in Sweden, Norway, USA and in the Netherlands (Cosic, Endersen, Foss & Palmqvist, 2013). The study of all these systems gave us a broader view of the possibilities and it has helped us to find a number of workable solutions and tools to design a prototype of a GIS system, which we find can be satisfactory to the DERS. Our focus is to help solve the DERS' tasks more efficiently. We aim to development a prototype software system,

[28] http://www.dmi.dk/vejr/tjenester/varsler/
[29] http://www.dmi.dk/en/service-menu/about-weather-warnings/

which will support and improve the communication between the emergency response teams using geographic information in an emergency situation. In this chapter we describe the basics about our programming techniques, how we design with HTML5 and how we have used the different programming languages, and got them to work together to build our web application. We have used the 3 basic languages HyperText Markup Language (HTML), Cascading Style Sheets (CSS) and JavaScript (JS). Our reasoning concerning the use of these 3 basic languages for web pages can be found in the following pages.

The global digital wave we are currently experiencing, created a social transformation from an industrialized (I) society to a digitized (D) society. A change from I-society to D-society, which both challenge and affects the individual's way of life and work, and the way in which we organize ourselves, educate ourselves and administrate the community. There is a large potential for the use of mobile applications as a platform in e-Government, and today there are more opportunities than ever before, especially when it comes to the use of spatial data. Mobile geolocation applications have been growing in popularity since 2004, with the development of one of the first modern social media applications: Yelp.[30] Since that time, there has been a huge growth in this market. From network and hardware applications direct from the providers; to Software Development Kits (SDKs) developed by software vendors. This has led to the plethora of solutions available today. Cloud solutions are certainly gaining appeal as well—Esri's collaboration with Amazon Web Services starting in February 2010 is proof of this (Holdener III, 2011).

In this chapter we aim to identify the scope of how HTML5 is designed, and how it can be used as a geolocation tool and a location-sharing tool in Danish Emergency Rescue Services (DERS). Before we started writing the code for our web application, we explored and evaluated different web based applications platforms like ESRI ArcGIS API (Application Programming Interface), Google API, Sencha, Leaflet, CartoDB. The ArcGIS platform includes APIs and software development kit (SDK or 'devkit') a set of software developments tools that allows for the creation of applications of an ESRI package.

### 5.4.1 ESRI platform

Our primary interest was concerned how to operate various tools and services in ArcGIS API, e.g. geolocation, WMS, WFS services and WKT format for representing layers and vector objects on the map. We were also interested in knowing how the available opportunities in ArcGIS can be used to build a web application for planning, and a communication tool for the DERS. To improve this possibility we experimented with two web applications; including address search, draw and measurement tools (based on ESRI platform) in our try outs. Figure 5-17 resembles an illustration from our experiments.

[30] http://www.yelp.com/


Figure 5-17: Screenshot from draw and measurement functions based on an ESRI platform (MTM group 4)

The Google Maps API provides web services as an interface for requesting Map API data from external services and using them within our own built map application. The Maps API Web Services are a collection of HTTP interfaces to Google services providing geographic data for web application. A list of services is located below:[31]

- Directions API
- Distance Matrix API
- Elevation API
- Geocoding API
- Time Zone API
- Places API

These services are designed to be used in conjunction with a map and it is possible to perform up to 100 queries per 24 hours free of charge. The other option is Google API for business, an application which must identify itself every time it sends a request to Google including an API key with each request. The API key is included as the value of a key parameter in the request URI.

[31] https://developers.google.com/maps/

### 5.4.2 Using the Google platform

We have tried to create the API based on the Google platform, the result is visible in Figure 5-18.


Figure 5-18: Screen-dump from WMS overlay application based on a Google platform (MTM group 4)

The background map is Google Maps JS API v3, and the code looks like this:

```
var map; var SDLLayer;  function initialize() { var center_pt = new google.maps.LatLng(55.668,12.545);
var mapOptions = {zoom: 13 center: center_pt };
map = new google.maps.Map(document.getElementById('map_canvas'), mapOptions);
map.setMapTypeId(google.maps.MapTypeId.ROADMAP);
```

and of course with regards to the Google JavaScript library in HTML:

```
<script type='text/javascript' src='http://maps.google.com/maps/api/js?sensor=true'></script>
```

Variable var SDL (Spatial Data Layers) Layer is used like JavaScript arguments to make a WMS overlay over Google Maps. WMS overlays are added from our server; which we have created for this purpose and with help of a JavaScript scripting language, the code is shown here:

```
function AddWMSLayermap() { //Define custom WMS tiled layer SDLLayer = new
google.maps.ImageMapType( getTileUrl: function (coord, zoom) {
```

and request to the GeoServer:

```
var url = 'http://mtm-gruppe4.no-ip.org:8080/GeoServer/gwc/service/gmaps?layers=MTM:lavbebyg_po&'
+
'zoom=' + zoom + '&x=' + coord.x + '&y=' + coord.y + '&format=image/png'; return url; // return URL for the
tile },
```

WMS overlay is used in the project application for visualizing building a GIS layer, and a HTTP interface for requesting georeferenced map images from a GeoServer. The interface is defined

with Cascading Style Sheets (CSS) where we have specified panel and form, size and colours of buttons. CSS panel code is shown below:

```
#panel { width: 180px; height: 100%; font-family: Arial, sans-serif; font-size: 10px; float: right; margin: 4px; background: #4e4949;
```

The next thing which we attempted on the Google platform was the drawing function to make a vector layer and subsequent data editing and intersection in the database with the aim that others may see changes in the GIS layer, and thus send a situational image of the focal area to other participants of the incident. A screen capture is shown below in Figure 5-19:

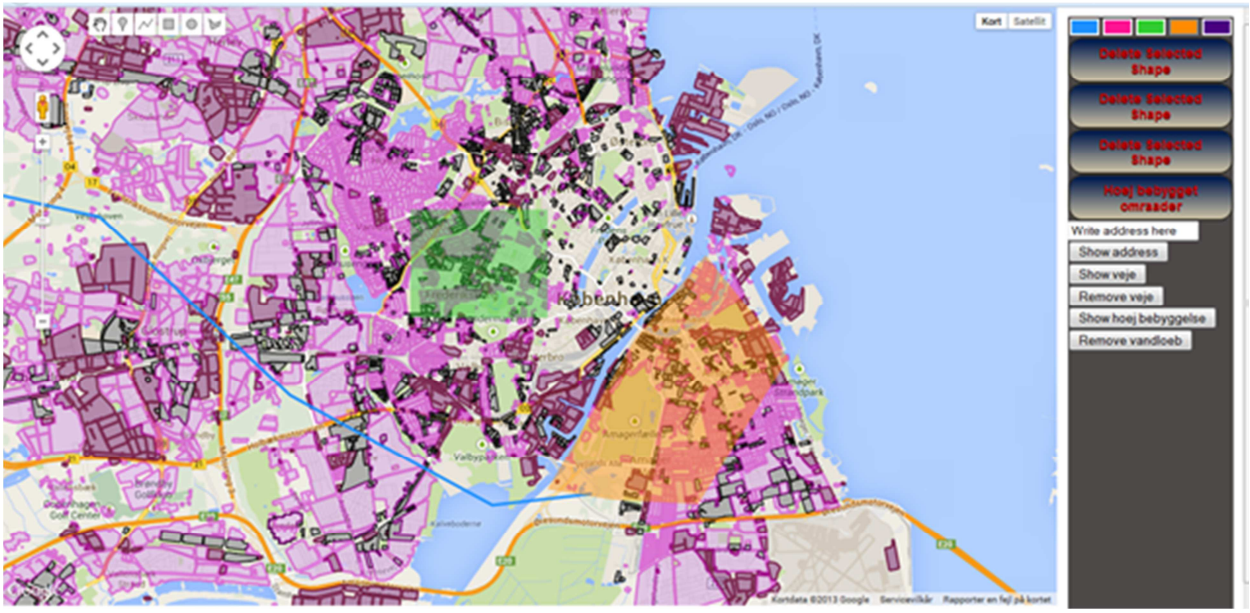



Figure 5-19: Screen-dump with WMS overlay and drawing function based on the Google platform (MTM group 4)

In addition to the ESRI og Google platform we would like to try some other platforms, namely Open Source platforms.

### 5.4.3 Sencha and Leaflet platforms

First we looked at Sencha products, primary Sencha Touch product suite that provides mobile developers with the framework and tools they need to build touch-based apps in a single integrated package. The next product was Open Source library. In Open Source library the interesting thing were; the Leaflet Mobile-friendly interactive maps. It works efficiently across all major desktops and mobile platforms, taking advantage of HTML5 and CSS3. Leaflet is meant to be as lightweight, and focuses on the features and easy ways to extend its functions with third-party-plug-in like layers and overlays.

We have divided the interface into four parts:

1. The main Page with an overview of Open Street Maps and geolocation.
2. The part of the layers; where the idea was to use plug-in in the form of WMS overlays from the GeoServer.
3. Search; with the possibility to search for addresses.
4. Different maps from Google, ESRI etc. e.g. topographic maps.

A screenshot of the web application based on the Leaflet platform is shown on the Figure 5-20.

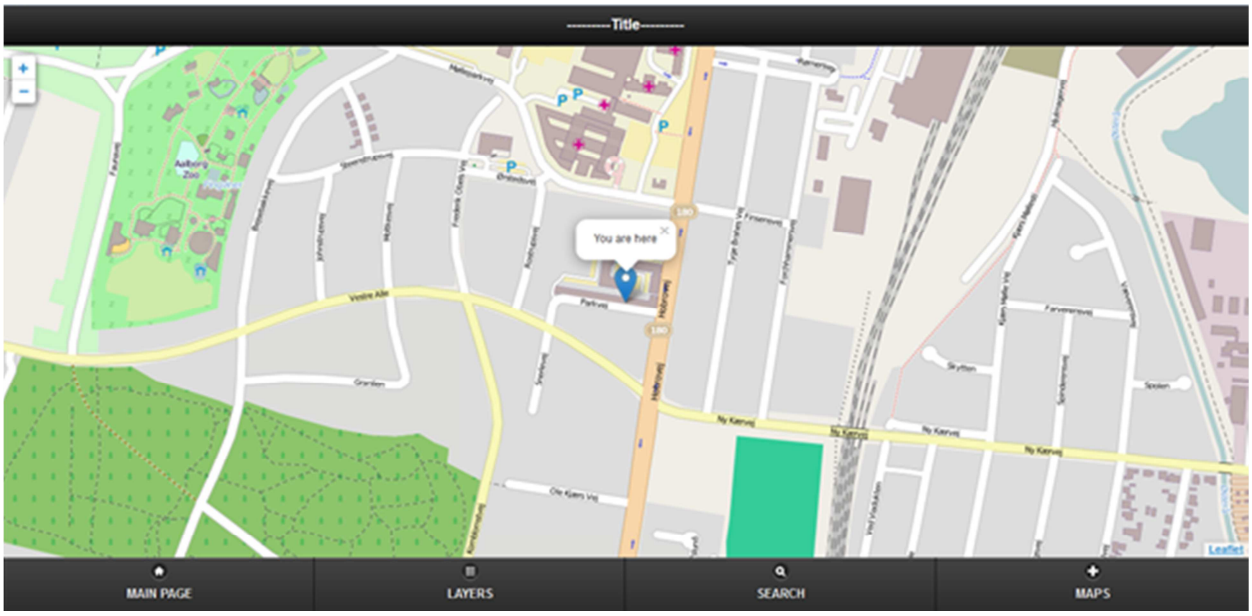



Figure 5-20: Screenshot web app based on the leaflet platform (MTM group 4)

### 5.4.4 Our application: 'GERT', Geographic Emergency Rescue Tool

Web applications are becoming more complex every day. During this project we have learned that the large GIS platforms, like ESRI and Google, offer a lot of web services and predefined web applications which are efficient and easily to use. But using the applications; ties the user to their system, and the application will be depended on the Google, Leaflet or ESRI servers. The user will not have the possibility to influence or change settings in the event that something goes wrong.

We have decided that it is not to our advantage to build a web application for emergency situations, which is dependent on other's services and support. Therefore we have decided to build our own web application independent of the large companies. Our system is a web application handling data from Kortforsyningen and data downloaded to our own server; address, buildings, roads, industry. Figure 5-21 illustrates the set-up of a user interacting with the application by a web-browser on a desktop or tablet. The application is built in HTML, CSS and JavaScript. Data is accessed using a PostgreSQL-database in cooperation with a GeoServer, which delivers WMS overlays to be shown in the application using an OpenLayers engine.

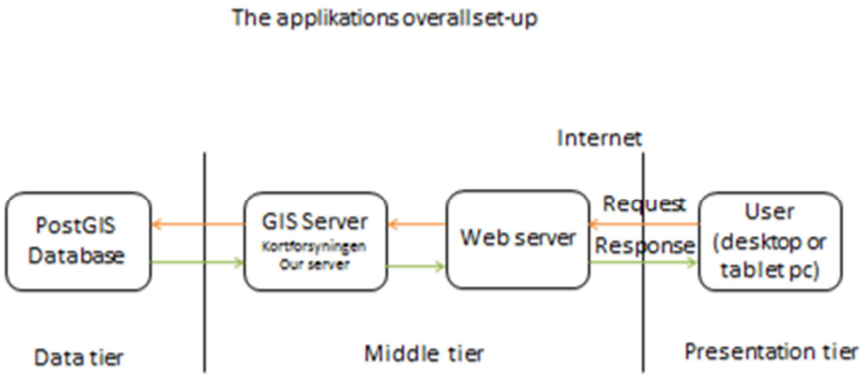



Figure 5-21: Set-up of our web application (MTM Group 4)

The application is built around a base map that is the main focus throughout the application with the possibility to select between a tempered topographic map and a satellite map. At the edges there are different toolbars containing drawing tools, maps and added layer tools, navigations tools, modify and measure tools, look at Figure 5-22.

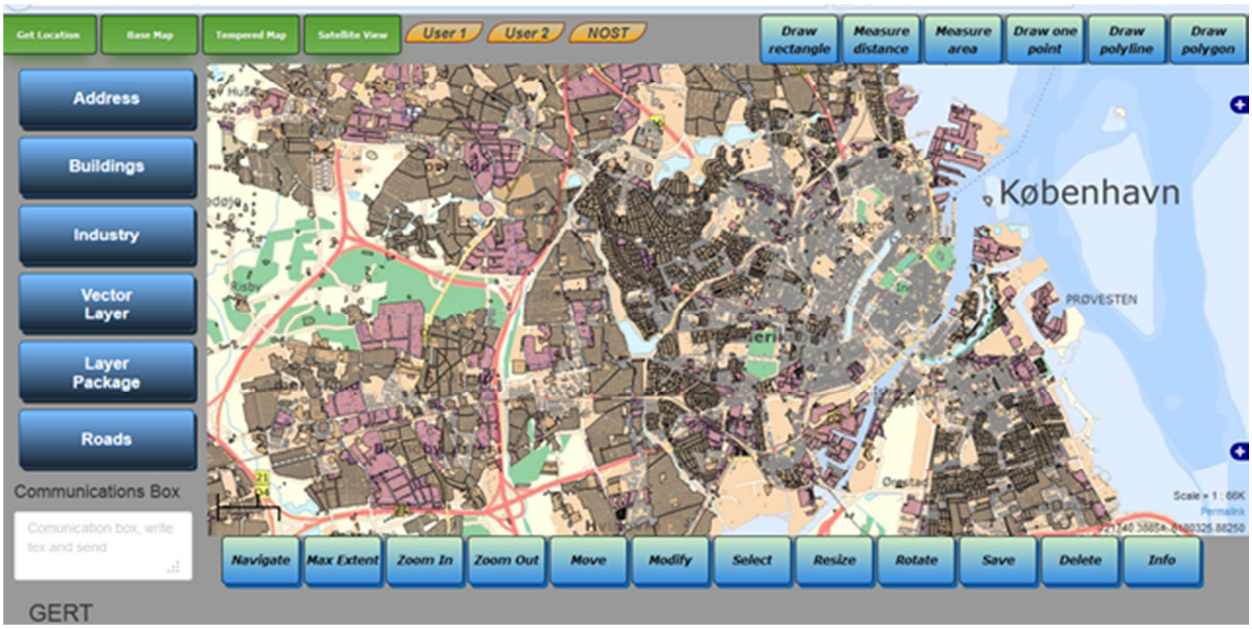



Figure 5-22: Applications home screen (MTM group 4)

Our web application is built with HTML5 which is the newest standard for HTML and it is a cross-platform application with the possibility to be used on a PC, tablet or smartphone. The application should be able to be used and still work without internet connection, and this is where HTML5's new offline storage comes in. HTML5 define a way to store files in a cache, so that when the user goes offline, the browser still has access to the necessary files. In our application that is CSS, HTML and JavaScript files. The source code is shown below. In HTML5 there is only one document type declaration, and it is very simple:

```
<!DOCTYPE html>
<html manifest='appcache.manifest'>
```

Defining the manifest file, resides on the server and dictates which file should be stored client-side in the browser's AppCache. In situations where the user goes offline, every manifest file has to start with CACHE MANIFEST. Then a list of the files, which we want to be stored and made available for offline use, is made.

```
CACHE MANIFEST
#section header to explicitly declare the following three files.
bootstrap.css   OpenLayers.js   index.html
```

The manifest file has to have the correct MIME Type, which is text/cache-manifest, with an extension for the manifest file and is added in the .htaccess file on the server.

One of the great things about HTML is that we have access to change how the cache behaves. We have access to the current state of the applications cache, and have function to asynchronously update it as well (Dixit, 2010). An example of JavaScript is shown below where we check whether the browser supports cache:

```
onload = function(e)  if (!window.applicationCache) {
 appCacheLog('HTML5 offline web applications (ApplicationCache) are not supported in your browser.');
 return; }appCacheLog('Initial AppCache status: ' + showCacheStatus(window.applicationCache.status));
```

When the page loads, set the status to online or offline:

```
function loadDemo() {
 // Log timing   if (navigator.onLine) {  appCacheLog('Initial online status: online'); return;
                  } else { appCacheLog('Initial online status: offline'); return;}
```

We then convert applicationCache status codes into messages:

```
showCacheStatus = function(n) {
                statusMessages = ['Uncached','Idle','Checking','Downloading','Update Ready','Obsolete'];
          return statusMessages[n];
```

To define the look and feel of our web application we have used Cascading Style Sheets (CSS) language. With our application design, we have tried to making a touch user interface with large finger-friendly buttons, taking account the elements described in the section of theories and methods. Similarly, there has been consideration of the map position compared to controls and buttons.

First we added a <style> tag to the HTML heading. The <style> allows us to add a *style sheet* directly on the HTML code and set the style's type as text/css. The first tag we defined was the body, see code below:

```
<style type='text/css'>
                body, html {position: absolute;height: 100%;width: 100%;
                                overflow:hidden;
                }
```

Changing the body tag will affect the entire visible part of page; therefore it is a good place to start.

We used squiggly braces { } to enclose the position rules for these styles. For each style we need to enclose all the rules in pairs of braces. Every rule consists of an attribute and the value which we want to give that attribute. An attribute name must end with a colon (:) and each value with a semicolon (;). If one tag has a lot of rules, the semicolon helps the browser separate all the various rules from each other. An example for this is our definition of a button, the code is shown below:

```
drawButton {
		-moz-box-shadow: 3px 4px 0px 0px #1564ad;
		-webkit-box-shadow: 3px 4px 0px 0px #1564ad;
		box-shadow: 3px 4px 0px 0px #1564ad;
		background:-webkit-gradient(linear, left top, left bottom, color-stop(0.05, #c9e8c3), color-
stop(1, #378de5));
		background:-moz-linear-gradient(top, #c9e8c3 5%, #378de5 100%);
		background:-webkit-linear-gradient(top, #c9e8c3 5%, #378de5 100%);
		background:-o-linear-gradient(top, #c9e8c3 5%, #378de5 100%);
		background:-ms-linear-gradient(top, #c9e8c3 5%, #378de5 100%);
		background:linear-gradient(to bottom, #c9e8c3 5%, #378de5 100%);
		filter:progid:DXImageTransform.Microsoft.gradient(startColorstr='#c9e8c3',
endColorstr='#378de5',GradientType=0);
		background-color:#c9e8c3;
		-moz-border-radius:5px;
		-webkit-border-radius:5px;
		border-radius:5px;
		border:1px solid #337bc4;
		display:inline-block;
		cursor:pointer;
		color:#0d0c0d;
		font-family:Verdana;
		font-size:10px;
		font-weight:bold;
		font-style:italic;
		padding:1px 1px;
		text-decoration:none;
		text-shadow:0px 0px 0px #528ecc;
		height: 40px;
		width: 70px;
}
.drawButton:hover {
		background:-webkit-gradient(linear, left top, left bottom, color-stop(0.05, #378de5),
color-stop(1, #c9e8c3));
		background:-moz-linear-gradient(top, #378de5 5%, #c9e8c3 100%);
		background:-webkit-linear-gradient(top, #378de5 5%, #c9e8c3 100%);
		background:-o-linear-gradient(top, #378de5 5%, #c9e8c3 100%);
		background:-ms-linear-gradient(top, #378de5 5%, #c9e8c3 100%);
		background:linear-gradient(to bottom, #378de5 5%, #c9e8c3 100%);
		filter:progid:DXImageTransform.Microsoft.gradient(startColorstr='#378de5',
endColorstr='#c9e8c3',GradientType=0);
		background-color:#378de5;
}
```

```
.drawButton:active {
		position:relative;
		top:1px;
}
```

This is a tag with a multiple rules for the button; where both the background colours for the button, text, button hover and activation of button is defined. First, we use CSS3 for Mozilla and WebKit box shadows, one to create the slight highlight at the top of the button and another for the drop shadow underneath the button:

```
-moz-box-shadow: 3px 4px 0px 0px #1564ad;
-webkit-box-shadow: 3px 4px 0px 0px #1564ad;
box-shadow: 3px 4px 0px 0px #1564ad;
```

Next we added the background gradients for Mozilla and WebKit, with a solid colour for all other browsers:

```
background:-webkit-gradient(linear, left top, left bottom, color-stop(0.05, #c9e8c3), color-stop(1, #378de5));
background:-moz-linear-gradient(top, #c9e8c3 5%, #378de5 100%);
background:-webkit-linear-gradient(top, #c9e8c3 5%, #378de5 100%);
```

Instead of the default button colour, we have decided on a blue gradient; which is defined with the start *colour hex code #378de5* and the end *colour hex code #c9e8c3.* The gradient is indicated in percentage, and in our case it is set to 5%. The colour gradient between the object's background and content is defined with the filter:progid tag element:

```
filter:progid:DXImageTransform.Microsoft.gradient(startColorstr='#c9e8c3', endColorstr='#378de5',GradientType=0);
```

In addition to button colour, the button border is also defined with radius and thickness in pixels and dark blue colour, hex code is #7a8eb9. Part of the CSS code is shown below:

```
border-radius:5px;
border:1px solid #337bc4;
```

Finally, the text declaration is defined with position, form, colour, font-family and font-size.

```
display:inline-block;cursor:pointer;color:#0d0c0d;font-family:Verdana;font-size:10px;font-weight:bold;font-style:italic;padding:1px 1px;text-decoration:none;text-shadow:0px 0px 0px #528ecc;
```

Hover selector is used to select the buttons when we touch over them. The term 'designated' refers to the process during which the cursor is hovered over the button generated by the element:

```
.drawButton:hover {
		background:-webkit-gradient(linear, left top, left bottom, color-stop(0.05, #378de5), color-stop(1, #c9e8c3));
		background:-moz-linear-gradient(top, #378de5 5%, #c9e8c3 100%);
		background:-webkit-linear-gradient(top, #378de5 5%, #c9e8c3 100%);
		background:-o-linear-gradient(top, #378de5 5%, #c9e8c3 100%);
		background:-ms-linear-gradient(top, #378de5 5%, #c9e8c3 100%);
```

```
                background:linear-gradient(to bottom, #378de5 5%, #c9e8c3 100%);
                filter:progid:DXImageTransform.Microsoft.gradient(startColorstr='#378de5',
endColorstr='#c9e8c3',GradientType=0);
                background-color:#378de5;
}
```

As in rules for button, here we have the same elements for definition of background, gradient and position with different colour values. The active state; where we want to imagine that we‘re depressing the button into the application. We achieve this effect by adding the following code:

```
.drawButton:active {
                position:relative;
                top:1px;
                }
```

Finally the markup for the button looks like this:

```
<button id='drawButton' onclick='javascript:setTool('line',this);'>Draw polygon</button>
```

We activate our button with an onclick function from javascript and add button id, the result is shown below:


We have made three different types of buttons for use in our application, but the procedure is the same as described above.

Now we are going to see how we used JavaScript scripting and the language and functions build in JavaScript. The javascript programming code, that we have inserted into HTML uses the following statement and is executed while the application loads:

```
                <script type='text/javascript'>
```

First we defined the background map from Kortforsyningen.dk and the javascript function looks like this:

```
var map;
                function init() {
                                map = new OpenLayers.Map('map', {div: 'map',projection: 'EPSG:25832',
                                                maxExtent: new OpenLayers.Bounds(120000.0, 5661139.2,
1000000, 6500000.0),
                                                maxResolution: ‘750.2',units: 'm',
var basemap = new OpenLayers.Layer.WMS( 'GST - skaermkort',
'http://kortforsyningen.dk/service?servicename=topo_skaermkort&login=&password=&SRS=EPSG:25832
',
          {layers: 'dtk_skaermkort'},
         {isBaseLayer: true}
```

We specified the map projection to EPSG: 25832 because the map from Kortforsyningen is in a format which requires coordinate transformation. Transformation of point coordinates from one coordinate system to another and we have done it with the help of proj4js library.

```
<script type='text/javascript' src='proj4js.js'></script>
```

Enabling these transformations in the browser allows geographic data stored in different projections; to be combined in browser-based web mapping applications.[32] Proj4js file is downloaded from Proj4js website.

Then we add variable dtk_skærmkort as WMS overlay with regard to the Uniform Resource Locator (URL) to Kortforsyningen where the map is located:

```
'http://kortforsyningen.dk/service?servicename=topo_skaermkort&login=&password=&SRS=EPSG:25832
',
```

In the same way we have added two more maps, namely dtk_skaermkort_daempet and ortofoto. All the base maps will run with the OpenLayer 'engine', but with information provided from Kortforsyningen and the maps themselves will vary.

We created a map but without any controls. It's not possible to zoom in, zoom out or navigate, therefore we add controls to a map. There are two methods for adding controls to a map:

1. Pass in a JavaScript array of OpenLayers Control object; when instantiating the map object.
2. Add controls to the map object after it has been created; by calling either of the two map functions *addControl(),* passing in an array of control objects.(E. Hazzard)

First we created navigations controls, zoom in and zoom out and in the same way add we scale line, mouse position, map overview and layer switcher. JavaScript code is written inside our init () function and looks like this:

```
controls: [
new OpenLayers.Control.Scale(), new OpenLayers.Control.ScaleLine(), new
OpenLayers.Control.MousePosition(), new OpenLayers.Control.KeyboardDefaults(),new
OpenLayers.Control.ZoomIn(),new OpenLayers.Control.ZoomOut(), new
OpenLayers.Control.OverviewMap(), new OpenLayers.Control.LayerSwitcher({'ascending':false}), new
OpenLayers.Control.Permalink('permalink'),
```

Afterwards we made a 'draw control' with Button Control class and our own custom control. One of the main uses of this function is that we usually pass it when we instantiate the button object. This way, we have made 'draw controls' for point, line, polygon and rectangle with the following code:

```
drawControls =
point: new OpenLayers.Control.DrawFeature(vector_layer, OpenLayers.Handler.Point),
line: new OpenLayers.Control.DrawFeature(vector_layer, OpenLayers.Handler.Path),
polygon: new OpenLayers.Control.DrawFeature(vector_layer, OpenLayers.Handler.Polygon),
box: new OpenLayers.Control.DrawFeature(vector_layer, OpenLayers.Handler.RegularPolygon,
                {handlerOptions: { sides: 4,irregular: true}
```

And select, delete, modify, rotate and resize button with following JavaScript code:

[32] http://trac.osgeo.org/proj4js/

```
select: new OpenLayers.Control.SelectFeature( vector_layer,{clickout: true, toggle: true,multiple: false,}
deleteTool: new OpenLayers.Control.SelectFeature(vector_layer,{clickout: false, toggle: false,}
modify: new OpenLayers.Control.ModifyFeature(vector_layer,{clickout: false,toggle: true,deleteCodes:
[46, 68, 27], // '46 is delete code properly turned into an array','68 is default deleteCodes include 'd"
rotate: new OpenLayers.Control.ModifyFeature(vector_layer, {clickout: false, resize: new
OpenLayers.Control.ModifyFeature(vector_layer,{clickout: false,}
drag: new OpenLayers.Control.ModifyFeature(vector_layer, {clickout: false,}
```

So, with the previous example, we created a draw button control and after that we created ‘measure line’ and ‘measure area’ tools with following function written in JavaScript:

```
function handleMeasurements(event) {
                var geometry = event.geometry;
                var units = event.units;
                var order = event.order;
                var measureTotal = event.measure;
                var out = "";
// because of the transformation units no longer matches the outcome of getArea and getLength, those
are in layer units.
                if(order == 1) {
                  if (units=='km'){
                                out += "Distance: " + (geometry.getLength()/1000).toFixed(0) + " km";
                                 }else{
                                out += "Distance: " + geometry.getLength().toFixed(0) + " m"; }
                                } else {
                                 if (units=='km'){
                                 out += "Area: " + (geometry.getArea()/1000000).toFixed(0) + "
km<sup>2</sup>";
                                 }else{
                                out += "Area: " + geometry.getArea().toFixed(0) + " m<sup>2</sup>";
```

Then we added address GIS-layer from our server as new WMS layer and add a blank background map that will advance the display of overlay data. The code for controlling the background map is made with inspiration and help from OpenLayers Cookbook and OpenLayers Beginner’s Guide. OpenLayers WMS overlay code looks like this:

```
 var address = new OpenLayers.Layer.WMS(
          'wms:addresse', 'http://mtm-gruppe4.no-ip.org:8080/GeoServer/wms', {layers: 'wms:adresse',
transparent: true}, {isBaseLayer: false}
```

To give the user the ability to draw and modify vector layers; we are adding a layer for editing called a vector layer, using the code below:

```
  var vector_layer = new OpenLayers.Layer.Vector('Vector Layer');
```

In the application menu there is a function called Get Location which centres the map based on the respective users’ current location. The following code shows the script behind get location function:

```
function geolocate() {
```

```
                        if (navigator.geolocation)
{navigator.geolocation.getCurrentPosition(showLocation,showError,{maximumAge:0, timeout:7000,
enableHighAccuracy: true});
                        } else {
                                alert('Geolocation is not supported');
                        }
        }
        function showLocation(location) {
                        var lonlat = (location.coords.longitude).toFixed(7) + ',' +
(location.coords.latitude).toFixed(7);
                        var coordArray = lonlat.split(','); lon = coordArray[0];lat = coordArray[1];
                        var proj_4326 = new Proj4js.Proj('EPSG:4326');
                        var proj_25832 = new Proj4js.Proj('EPSG:25832');
                        var nyproj = new Proj4js.Point(lon,lat); Proj4js.transform(proj_4326,
proj_25832, nyproj);
                        var center = new OpenLayers.Geometry.Point(nyproj.x,nyproj.y);
                        map.setCenter(new OpenLayers.LonLat(center.x,center.y),11);
                        x = center.x; x = x.toFixed(0); y = center.y; y = y.toFixed(0);
                        accuracy_xy = (location.coords.accuracy).toFixed(1);
                        map.layers[4].removeAllFeatures([feature_user_location]);
                        var user_location = new OpenLayers.Geometry.Point(center.x,center.y);
                        var feature_user_location = new
OpenLayers.Feature.Vector(user_location);
                        var user_location_accuracy =
OpenLayers.Geometry.Polygon.createRegularPolygon(
                                user_location, accuracy_xy/9,40,0);
```

As seen earlier, geolocation uses the class navigator.geolocation.getCurrentPosition, which is used to retrieve information on the users' location. This initiates an asynchronous request to detect the user's position, and queries the positioning hardware to get up-to-date information. When the position is determined, the defined call-back function is executed.

Measuring tools is a very useful feature and it is of course added in the our application. It is possible to choose the distance measurement tool, in our application it is called 'measure distance' and 'measure an area'. It is easy to calculate the distance from one location to another in the base map; just choose the measure distance button and then click on the desired location on the base map. The other measuring tool is measure area; zoom and pan the map to find the area of interest, and then place vertices for the polyline to define polygon. The area enclosed will then be outputted in square meters or square kilometres. JavaScript code used for the definition a tools is shown down below:

```
function handleMeasurements(event) {
                                var geometry = event.geometry;
                                var units = event.units;
                                var order = event.order;
                                var measureTotal = event.measure;
                                var out = "";
                                // because of the transformation units no longer matches the
outcome of getArea and getLength, those are in layer units.
        if(order == 1) {
```

```
                if (units=='km' out += "Distance: " + (geometry.getLength()/1000).toFixed(0) + " km";
                 }else{          out += "Distance: " + geometry.getLength().toFixed(0) + " m"; }
               } else { if (units=='km'){out += "Area: " + (geometry.getArea()/1000000).toFixed(0) + "
km<sup>2</sup>";
        }else{ out += "Area: " + geometry.getArea().toFixed(0) + " m<sup>2</sup>";}
               message = 'Measured: ' + measureTotal.toFixed(0) + " " + units;
               changeMessage(message);
```

That was a short description of our web application and the structure use, with focus on the technical programming details. In the next chapter we evaluate our prototype in terms of being a rescue and warning system.

## 5.5 Our prototype as a rescue and warning system

One might consider whether the recorded information under an emergency situation also could be used in a wider forum, in order to warn people and save values. This could be achieved by sharing information with other authorities and citizens by providing information on, for example. blocked access roads and the risk of loss of values due to flooding. This information could possibly help ensuring better access roads in the affected areas and potentially minimize loss of values. So who should be warned? It would be obvious at first hand only to think of residents and other parties with interested in the affected area. But also external influences should be examined, for example whether a redirection of public bus and other transport that would otherwise pass in the affected area could ease the rescue work. A map is a visual powerful tool that can give you a quick orientation in an area than, for example, a list of street names define an area. local media and local authorities would be appropriate channels for the distribution of information to affected areas through their websites and radio.

## 5.6 Performed software testing

The aim of our testing is to develop an application, which can run a series of functions according to our specifications. During the development we will continuously run our own testing, so called white-box testing. Where we develop and perform the accompanying test as soon as possible to ensure success. An example of white-box testing can be to implement a map view to a web page; the code is written in the an HTML file and it opens the.html file in a browser. The result will then be either a new map view or an error. In case of error the developer returns to the code and should try again until a map view is visible in the browser. We also test each other's results, this we call Gray-Box testing. As we all have an insight as to the code behind the individual function, but we have not developed all the functions which we will test. Black_Box testing is run in the final stages of implementation. We invite possible users and others outside to test our software. Some testers are coached others not.

| Tester | Test ID | Error description | Error location | Description of the correction | Problem solver |
|---|---|---|---|---|---|
| Azad | A002 | Simple polygon is uploaded as multipolygon in database | Database | changing the options in database | Azad |
| Azad | A002 | GeoServer cannot start | GeoServer | Lack of a file in installation | Azad |
| Bjarke | B003 | test service ports closed | Router | Bridge mode (provider) + new router | Bjarke |
| Bjarke | B004 | TomCat/GeoServer WAR | WAR size | OpenGeo | Group |

Table 5-2: Selection of errors, descriptions of the errors and who found the error and who solved the error (MTM group 4)

To keep track of the errors which we experience, a table for error descriptions has been created. Table 5-2 is a sample of the more comprehensive table in appendix 13.

### 5.6.1 Testing functions

All buttons for layers and functions must be tested to ensure their accuracy, both in an order and in a disorderly manner. All errors found in both the orderly and disorderly testing should be listed in our table with error descriptions.

#### Orderly testing of functionalities

When testing our functionalities in an orderly manner, the success scenarios from our use cases must be followed. This will give either positive or negative results. The orderly testing resembles the actual use of our software in a real life-situation.

Examples of our functionality testing

- Does the zoom buttons zoom?
- Can I pan in the map view?
- Etc.

#### Disorderly testing of functionalities

Using the software in an erratic manner e.g. monkey test. This type of testing is meant to 'confuse' and provoke the software to induce errors.

### 5.6.2 Testing system capabilities

When testing the application, the purpose is to stress the system to the maximum. How far can we take the testing?

- How many layers can the list contain?
- Can I pan all the way to America and then zoom back into Denmark using the 'Home' button?
- Can I go to the main square in Copenhagen with my tablet and use this software?
- Will I be able to see on my tablet where the other field officers are located in my vicinity?

These questions and many more comprise the basis of our testing format. Testing the capabilities of the system will also help to identify when or where the system fails.

#### Examples of our capabilities testing

### 5.6.3 Analysing our test results

Our application is built for use on a pc and tablet. We have tested successfully that our software can run alongside other applications. It is possible to work with different software programs at the same time and it is an advantage because it allows the user to work with multiple applications at the same time. This functionality is tested and works well enough.

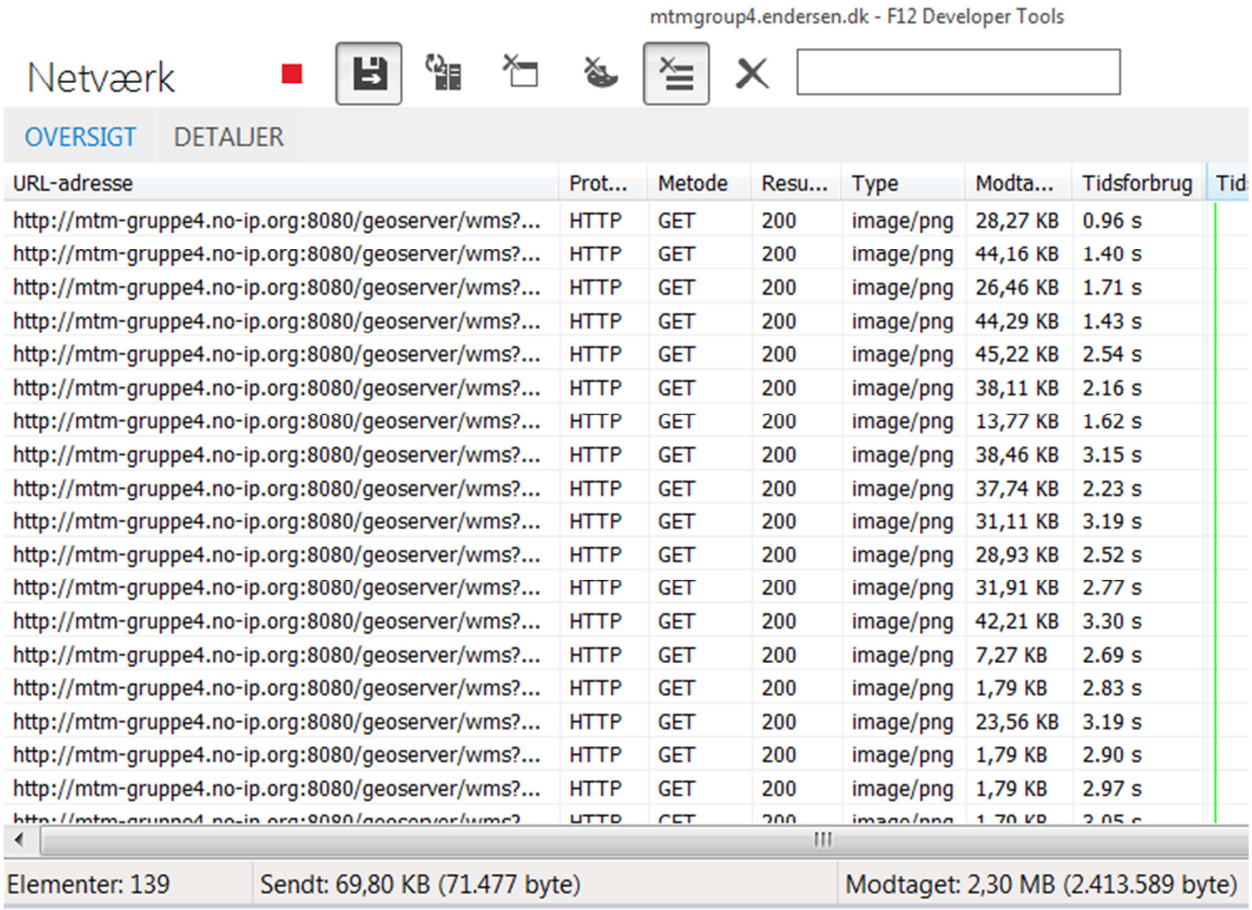
Multi-touch is a term that refers to systems that allow more than one input at the time. If a system supports multi-touch, it is possible to also support more advanced use, such as two fingers apart to zoom in or four fingers across the screen to change a screens view. This function has been tested in our application and works fine.

Most modern smartphones and tablets have built-in gyroscope, which enables them to change the layout according to their orientation. Screen size and orientation are perhaps the two biggest challenges when designing user interfaces for mobile devices, and it is essential that this is taken into consideration. In our application the graphic elements work well on a large screen in landscape orientation and working on a small screen or using portrait orientation is less successful.

Testing the application; by using different browsers: IE, FF, CHROME, and SAFARI.

The application for a pc has been tested using 4 different browsers; Internet Explorer (IE) version 11, FireFox (FF) version 26.0, Chrome version 31.0.1650.63 and Safari version 5.1.7. FF resembles IE and Safari resembles Chrome in appearance. When using a high screen resolution we recommend FF or Safari. On bigger screens, we have found that the use of low resolution is an advantage to the visual appearance of the application.

The load time of a web application or website is essential for the DERS as their goal is to save time. All of tested browsers have the ability to test how long it takes to load the application; in FF it is necessary to download Firebug, which is an extension to FF, to see it. In Figure 5-23 we will show how it looks like in IE. It shows the time for each element and forgets the total time we should export it to excel and get the sum in excel and compare them.




| URL-adresse | Prot... | Metode | Resu... | Type | Modta... | Tidsforbrug | Tid |
|---|---|---|---|---|---|---|---|
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 28,27 KB | 0.96 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 44,16 KB | 1.40 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 26,46 KB | 1.71 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 44,29 KB | 1.43 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 45,22 KB | 2.54 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 38,11 KB | 2.16 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 13,77 KB | 1.62 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 38,46 KB | 3.15 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 37,74 KB | 2.23 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 31,11 KB | 3.19 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 28,93 KB | 2.52 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 31,91 KB | 2.77 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 42,21 KB | 3.30 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 7,27 KB | 2.69 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 1,79 KB | 2.83 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 23,56 KB | 3.19 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 1,79 KB | 2.90 s | |
| http://mtm-gruppe4.no-ip.org:8080/geoserver/wms?... | HTTP | GET | 200 | image/png | 1,79 KB | 2.97 s | |



**Figure 5-23: Shows a print screen of IE which we can use to calculate the total time for opening our application and by doing the same in other browsers, we can find out which browser is faster (MTM group 4).**

As mentioned earlier, speed and performance is essential; that the user does not have to wait long for the application to load. Therefore we have also performed 'download speed tests' on the webpage test website, these test results are shown on the Figure 5-24:

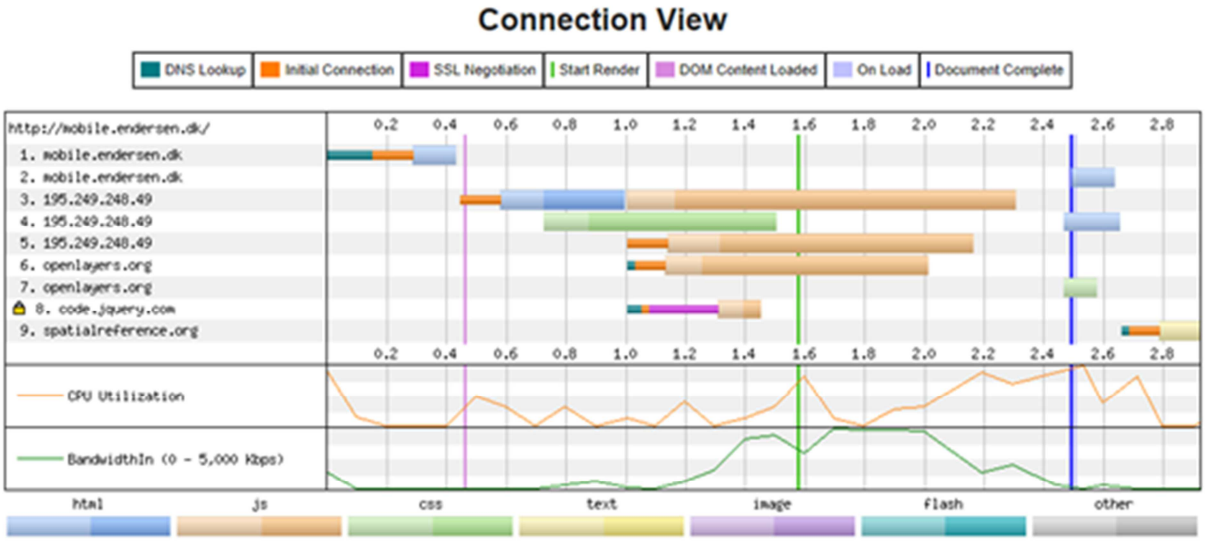



Figure 5-24: Testing the speed of downloading using our application. The test is performed using Webpage test (MTM group 4)[33]

The test show the time of accessing our application, which contains HTML, JavaScript, CSS, text and an image.

Software accuracy in the field, gps

Localization, geodata referencing is most important part in the work with spatial data. In this context, knowledge of determination accuracy is extremely important information, which is often crucial for the further uses of data. The project application use two different methods of positioning, and it is therefore natural to investigate the accuracy of these methods is likely to be.

Applications Get Locate function with use Wi-Fi internet network and  measured with use two different browser, shows somewhat better accuracy and expected, approximately deviation is 15-20 m. The function does not have full access to the device, and accuracy is therefore based on the positions calculated from the networks master. The accuracy is very much depending on the presence of wireless networks.

The local application, with full access to the device's GPS functionality is as expected, with an average deviation of 10 meters.

The application is tested in different part of Denmark and tablet edition with Get Location function is tested in Aalborg and Løkken. Testing has shown zoom in, zoom out, pan and geolocation working well also when user is in other part of Denmark not only in working area.

Usability and user experience testing is vital to creating a successful web application. We have not done user experience testing but we have running our own tests to find out how users are interacting with our application and where problems is.

[33] www.webpagetest.com

We can determine that under convenient conditions the most functions in our application are working as intended. We have not had the possibilities of testing under unfavourable conditions of extreme rain or life critical situations.

No system is completely free of errors, especially not newly developed software. To achieve a virtually flawless system it will require comprehensive testing; both structured, unstructured and erratic testing, use of the product and continuously corrections and testing of the corrections for a very long time. Most systems today offers their users the possibility of reporting the software defects which their encounter; this can also be seen as a sign that even experienced software developers are aware that software system might not be 100 % perfect.

## 5.7 Summary of technology and programming

We have set up two pc's working as servers and discussed the challenges as they appeared. We have installed the OpenGeo suite solution, where we have created our own data layers and imported existing data into PostGIS. Through our GeoServer, we have displayed both WMS and WFS-T layers for external use through our web application. Two user interfaces have been designed, one for a tablet user and one for a pc user. We have discussed the users' need for accessing data, and assessed the requirements towards the use of equipment. We have explored and evaluated different web based applications platforms like ESRI ArcGIS API, Google API, Sencha, Leaflet, and CartoDB. We have decided to build an independent application in HTML, CSS, and JavaScript and have demonstrated how a button is defined in CSS. We have added a background map, layer from server and added drawing and function controls to the map. A brief evaluation of the prototype as a rescue and warning system has been assessed. We have in various ways tested our application on different devices, e.g. pc, web browsers, trying to detect and correct errors.

The overall aim of this chapter has been to answer 4 of our research questions in regards to our own work performed during the progress of the project. Just to freshen up on our research question they are listed here:

*How can we develop a prototype, a mobile application based on web standards and open source products, meeting the needs of the emergency rescue teams and NOST?*
We have developed a prototype, a mobile application based on web standards and open source products. As to whether our prototype met the needs of the emergency rescue teams and the NOST, we have to be honest: this is a prototype not a working tool which met the needs of the DERS users.

*How can we establish and host a database to support updating?*
We have established and we host databases which can display data to our users. But unfortunately we have not been able to achieve real time update as of this point in time.

*How can we design an appropriate user interface for emergency response work?*
We have designed 2 user interfaces providing our indoor and outdoor users with map views and functions. We believe that user intuitive and simple design is the path to follow.

*How can the developed prototype tool be used as a human rescue and warning system?*

Implemented in our prototype is the layer of Address points. By selecting from this layer it is our believe that population to be warned can be identified and subsequently be evacuated.

# 6 Discussion

In this chapter we evaluate our project, and discuss whether it is relevant and usable in regards to the Danish Emergency Rescue Services (DERS). We compare our ideas with other projects, concerning the use of GIS for communication, to illustrate the relevance of a geographic information tool for communications in DERS.

In the end of this chapter we discuss the question Is this the correct product for DERS personnel? This is a question to verify the existence of our kind of geographic communications tool for the DERS. We will also discuss and answer the question: '*Does this product work correctly?*' in order to evaluate validation of our product.

In general; one question should be asked to the data providers and that concerns the data quality and quantity of the data which they provide. An application will rarely be better than the quality and quantity of the data made accessible and used in the application. Adding value in an application is a possibility, but it is still limited by the input data. An example of why it is limited by the data quality can be shown in an experience in which members of our project group had at the GST workshop in the spring of 2013. A map of a local area of Lolland-Falster was visualized on the wall. On the map dots representing citizens; care homes, schools, nurseries, etc., all residents who would need aid in case of an evacuation. One man from the DERS looks up at the map, and asks: *'where are the dots representing the Day care centres?'* The representative from GST recognizes that the data on Day care centres are missing, and promises that the data will be introduced in the future. What if other datasets are missing or one or two of the care homes are not registered as dots? Maybe the dataset was last updated in December and the dataset is updated once a year, and new care homes have been opened in January. If an incident of Storm Surge occurs between January and December is it then acceptable that the data is missing from the datasets for 11 months? Is it the data providers or the users' responsibility to ensure that the data shown in the map is correct?

## 6.1 Our Application

Some of the issues which have been discussed a great deal internally in our group have been practical issues. Issues like how much information can we, within reason, include in our application without confusing and stressing the users? We have also questioned whether we should include a discussion on the actual use of the application in the field, or should the user use a pen, a special glove or bare fingers? How many zoom levels must we support? How many buttons do we need? How many buttons is there room for on the screen? Which Colours, Sizes, Naming, and Fonts?

Regarding zoom levels, our application has 15 zoom levels which are from the full extension, covering all of Denmark, into street level. One could argue that a maximum zoom level for zooming out could be set for all of Denmark, a region or maybe only a municipality or city. But should we then also limit the amount of input data to match? Our answer to this question is no. We advocate that neighbouring DERS teams should be able to assist each other, and be able to view the same data. Setting a limit to the zooming and the amount of data visible would

probably minimize the amount of data to be stored on the servers, but it would also limit the use of the software to specific areas. Imagine being in the maximum zoom level, viewing your own municipality, and needing to see the situation of your neighbouring municipality many kilometres away, that would result in a lot of panning or maybe even an extra button or layer to help you to view their focal area on screen.

How many buttons can be fitted on a screen of a tablet? How large must they be? How many do we need for our system to operate at maximum functionality? How many would actually be needed in an actual emergency situation? The size of the buttons must match the pen, finger or glove which is to be used to mark on the screen, which will set the size of the buttons to approximately 1 cm to 2 cm in diameter. A pen or a finger will be able to use a 1 cm in diameter button, while the hand wearing a glove will need the larger buttons. Comparing these button sizes; it will be possible to have up to 4 times as many buttons on the screen with the smaller buttons compared to the application with the larger buttons. Returning to the question of how many buttons actually are needed? For our prototype application we have minimized the number of functions compared to what a real life application would include. Comparing our required functions to the ones included in GeoConference (PCI, 2009) it is clear that our application only covers a very small part of the functions needed to fulfil the needs of a comprehensive system for the DERS teams.

In this report we have not found the ideal solution for the practical use of our application in the field in regards to pens, gloves and bare fingers. Is it a possibility for the emergency commander to remove his gloves to handle the tablet in an emergency situation? Can the emergency commander handle a pen wearing his gloves during an emergency situation? Can the standard gloves of the emergency personnel be used on a tablet? Can the emergency commander use special gloves during the emergency situation to use a tablet? To consider all these different practical methods it is also necessary to keep in mind the size of the 'tip' to be touching the screen; the bare finger and the pen are small compared to a glove. Small 'tips' can mark a button or point with great precision, while gloves are not so precise and require large buttons. Using a pen or bare fingers require the user to remove his or her glove(s), then what to do with the glove? Removing the gloves can possibly result in the loss of a glove. Pens are small they can get lost in the field or in the emergency vehicle. The use of special gloves for using tablets could be an entire project in it selves, like the project of Jihyun & Kyung: 'Electrical properties of conductive fabrics for operating capacitive touch screen displays' (Jihyun & Kyung, 2012), evaluating silver-plated nylon filaments as conductive materials in knitted gloves. Also there is an extensive (unscientific) evaluation of both knitted and leather gloves on the internet[34] where everybody can join in and evaluate the various glove types.

Regarding publishing of our data as WFS-T; which would give the rescue officers the possibility to add, edit, delete and update a feature, which can see in section 5.1.2 'GeoServer' under Figure 5-11. Test from Q-GIS has shown problem with updating with WFS-T data. We have access to our database by use of our GeoServer across the internet and can Add, Edit, Delete, update, the data e.g. 'Address Point' layer in GeoServer. We can conclude further work has to be done in order to have full working web service.

[34] http://www.touchscreenglovereviews.com/

In our application we can display our data as WMS, zoom, draw; lines, rectangles, polygons and points, we can select, modify and measure features. It is also possible to geolocate the user's own position and change between base maps.

## 6.2 Pros and Cons of Communication with Geographic Information

Using a Geographical Information System (GIS) as a communication system and a tool, is widely spread across the world both geographically and professionally. At many conferences and other professional meetings GIS is becoming a familiar term. For example at the National Conference on Health Communication, Marketing, & Media, there was an entry on Geographic Information Systems (GIS) Tools: Fostering Healthcare, Health Communication Activities and Materials by Rebecca Ledsky (Ledsky, 2013). Ledsky's conclusions were that *'GIS tools have practical applications in health communication – for planning, engaging, and creatively communicating with a wide variety of audiences.'*

The aims of our system containing geographic information, compared to a typical GIS system from for example ESRI and MapInfo, has been to make our system usable for people who are not experts in GIS. As described by Cai, Yu & Chen: *'A typical full-strength GIS system today utilizes a layer-based geographic information model for characterizing and describing our world, and is commonly packaged with hundreds of data processing tools and analytical functions. However, due to the incremental development of GIS from a diversity of application domains (management of census, land use planning, transportation analysis), the terminologies and metaphors used for exposing GIS functions to users are conceptually confusing and cognitively complex'* (Cai, Yu & Chen, 2013).

Other has previous experimented with the combination of Open Geospatial Standards, WFS-T, GPS, GeoServer, Mobile devices etc. as we have for this project (Engravslia et al., 2010). Engravslia et al. concluded that the use of standards and the combined efforts of many people made their project a success. We feel we have achieved a good working balance within our team and hope to have a working prototype when we present our project. Whether or not this happens we have learnt valuable lessons in project development and teamwork.

## 6.3 Verification and validation

Is this the correct product for DERS personnel?

At the moment it is too early to conclude the possible success or failure of the product. Our product is a partial solution and will require further development to be a fully functional solution in regards to the software needed by the emergency commanders at DERS. The principle behind the system should be able to support the emergency commanders and the NOST operators in times of emergency.

Does this product work correctly?

At one point in time we had functional solution, where it was possible to insert, modify and delete objects on top of a map in GeoExplorer a part of the OpenGeo solution. Unfortunately some later changes has affected our application setup, so that the functionalities of applying

changes has been lost and we hope to have the product up and running again in time for our presentation.

## 6.4 Next-generation 'The Future'.

For the Next-generation of our system, the implementation into a common tool at DERS will involve the users and their will to adapt. The users' must be introduced into both the benefits and the possible downfalls. Any unexpected downfalls will possibly influence the users' opinion in a negative direction.

The next-generation system should help the DERS and NOST to transfer from analogue to digital working methods, as this evolution represents a great untapped potential for efficiency and quality improvement (Engravslia et al., 2010).

Another possibility for the Next-generation system could be support in case of other emergencies than flooding, e.g. fire detection and firefighting, earthquakes or maybe terrorism. Fechner et al. presented at the GeoInfo 2010, Proceedings of 11th Brazilian Symposium on GeoInformatics 'Low-cost satellite-based products for the Web – the example of Fire Web Service'. Fechner et al. discusses the use of a satellite-based dissemination system and assessing fire products through the Fire Web Service, based on Open Source software.

To gain further information of emergency situations it would be useful to have access to photo coverage of a situation, similar to the FEMA Smartphone App, where it is possible for the public to take and submit GPS photo reports of disasters, these can then be displayed on a public map for others to view (FEMA, 2013).

- Public assistance
- current situation documented by emergency personnel
- Georeferenced images

An active system for the use of the DERS personnel will also have to be submitted to a highly sophisticated security system. Securing the system from hacking, malfunction, power shortages, flooding and fire. The stability of the system must be ensured to overcome ordinary technical difficulties such as the system going down due to a hard drive malfunctioning, this should solvable, basically by having a duplicate system (or 2 or 3 identical systems) always running as a backup system.

The Next-generation system would gain from including data from the Danish Environmental web-site Miljøportalen. Including data from Miljøportalen will support the emergency commanders and NOST on e.g. the location of explosives and chemicals in the local area. This knowledge is valuable in regards to safety, risk assessment and possibilities of pollution.

ESA (European Space Agency) are developing five missions called sentinels under Copernicus program. The mission must fulfil by constellation of two new satellites in the space for providing information. These satellites should be launched between 2014 and 2015. Sentinels 2 is a mission to providing the high resolution images for land for example imagery of soil and water

cover, coastal areas and waterways, Sentinel 2 will also provide information for rescue emergency services which NOST and DERS get benefit of that.[35]

For the ideal solution to be used by the DERS emergency commanders and the NOST users, a plan has to be made concerning the possibilities of an internet breakdown or power shortage affecting the internet masts. A great deal of consideration must go into defining the minimum requirements of an offline solution, how much data can be stored on directly on the tablet? Which data is life critical? What can be spared? How much 'uptime' is needed to update the tablet complete with the fellow users' status information? How long a 'downtime' is durable? Where do the boundaries of 'uptime' and 'downtime' lie compared to the emergency commanders in the field, the NOST users and finally compared to the developers and, regarding programmers, who is to create the comprehensive system. Offline applications for tracking movements on a map actually already exists. One of these is Couch Tracking (couchtools.com), this system can be set up to track while the app is offline and automatically share the logged events when gaining online access again. A function similar to this could be a good offline solution for the DERS.

A use function; which the DERS would see as a great benefit in cases of evacuation is the ability to identify the day population of an area. Today as most people have mobile phones, it should be possible to develop a tracking system for the DERS to identify all the mobile phones in within the evacuation area, and to send them all a SMS to warn them of the upcoming event and the evacuation. This idea was briefly discussed in chapter 5. To keep track of the day population if should be possible to visualize them all as dots on the map, and thereby be able to identify the dots not moving out of the evacuation zone, and offer assistance in case of need.

## 6.5 Source Analysis

In this section we want to evaluate those sources which we have used to answer our questions. Here we write generally about the sources and do not evaluate them all one by one. For evaluation of sources we consider the following points; credibility, up to date, bias and point of view and accuracy (W. D., 2007):

- **Creditability:** When relaying on a source; it is important to know who the author and the publisher are. As users of a source we have to be careful, if we cannot identify the author and the publisher. To rely on a source without knowing the author and his credentials is bad practice. We have tried to get our information from mainly primary sources; i.e. articles and technical books, and less secondary sources i.e. internet pages. The credibility of our sources differ, e.g. some topics are covered in many sources while others only have one reference source.
- **Up to Date:** How current is the sources? When referring to web sites and other online data; it is important to remember that these pages can be updated, deleted or moved at any time. We have sought to use scientific, recent and up-to-date sources to build our project upon. E.g. in this report there is a section describing the theories of GPS and the use; the theories have not changed for years therefore we find it acceptable to use a book from 2002 (Dueholm & Laurentzius, 2002); but when we touch the subject of the

[35] http://www.esa.int/Our_Activities/Observing_the_Earth/Copernicus/Overview4

accuracy of GPS today we use a source from the '*Official U.S. Government information about the Global Positioning System (GPS) and related topics*.[36],

- **Bias and point of view:** What is the intention of using specific sources? Is it to persuade, inform, or maybe sell? Does the web sites addressed with top level domains e.g. .com, .gov, .edu? Checking the credibility of a source help to determine any bias. For example; when we know the information about evacuation in an emergency situation is written by DERS, then we can determine that the purpose of information is just for information. We have been very cautious not to rely on a sole resource in our research. As this is a study project and not a project with commercial interest; we have been free to include the sources of our own choices.
- **Accuracy:** How accurate are our sources? This question confronts us with three additional questions:
    - Is the information free for grammatical, spelling and typing errors which can lead us to misunderstanding?
    - Are there any research study and/or statistical discussions?
    - Are there sources for all factual information? if so, can they be identified easily?

  We trust our primary sources to be accurate and current, they are all well written and of scientific quality. The secondary sources are more doubtful, but we still have confidence that these sources are usable.

## 6.6 Reflection

For this project we would have liked to include a user login identifications system, to handle both user preferences concerning the setup interface setup, e.g. left-handed user/right handed user. And also identifying the user; by an ID number for registering the users' moves in the system, and for securing the system against unwanted users.

During this project we have had to re-adjust our expectations and goals, regarding our prototype. We would have liked to create a large scale usable system for the DERS, but we have had to be realistic in our goal and time limits.

There are a couple of things we regret not having specified from the very first moment we started this project. We would have liked to have set the target low, so that we could build upon a minor solution. We made a mistake here and had to downsize a couple of times. It would properly also have helped us if we had had a clear and uniform vision of the solution from the beginning, in the form of e.g. a ER relational diagram or an Object diagram. It would also have been a surplus if we, from the start, had discussed our options concerning access methods to information. Where do we get the information from and how is the user supposed to access a specific information using our tool.

Finally we had a lot of trouble setting up the entire system from scratch. Setting up a GeoServer, collecting data and publishing them, etc. None of us have ever done all of these tasks before, but we felt this was to correct way to proceed as a comprehensive system for the DERS would have to follow many restrictions. Some of these restrictions are e.g. keeping the data in Denmark in case of a terrorist attack, using Danish data instead to Google data; as one

[36] http://www.gps.gov/systems/gps/space/

can never know when data suppliers like GOOGLE or OpenStreetmap decides to change their data.

For this project we would have liked to seek out users from the DERS to introduce them to our application or for them to try it out. We would also have liked to have time to take a session with some of the potential users from the DERS or NOST, to introduce and educate them on the potential use of our application. In a real life setting, starting up new users to a geographic communication system would include training; education in the use of the application. As the right usage of the system is one of the essential points under the rescue operation and it can be achieved through education. As we have attempted to minimized 'noise' in the user experience and aimed for the user interface to be intuitive, the amount of training needed should be minimal.

In chapter 5 a discussion of Compass roses was started. What kind of compass roses will the user of our application see when applying advantages or disadvantages? If the users feel confident and are accustomed to the top of the screen always representing north then the user might need a compass to help him or her turn around to be able to match the map to the surrounding. The required compass could be a handheld one or a small compass rose in the corner of the map view. Other users might be comfortable with the map being flexible, in a manner independently turning to match the magnetic north. With a GPS in the tablet all these ideas should be possible.

Concerning our prototype as a rescue and warning system, our prototype contains a selectable layer (address points) representing a homes of the residential population, we can identify the residents within our focal area and any other area of interest. Selecting all the address points within the area of interest, will help the field officers to e.g. evacuate all the local residents. It is expected that the NOST can enrich the selected addresses, with phone numbers, CPR data (the civil registration numbers) and the count of people living at the addresses. And also identify the residents listed as citizens with special needs, e.g. disabled, care home residents, kindergartens, schools, etc. In the ideal world; the first action of evacuation is to be the delivery of a text message (SMS) to all the mobile phone numbers identified as listed within the evacuation area. This should warn most of the residents in the area, resulting in the residents being able to transport themselves to the evacuation centres. This leaves a smaller number of residents to be aided in their evacuation by the DERS. Additionally some of the documented focal areas and evacuation areas can be used to inform the public using the media; TV and radio, and maybe Facebook, Twitter, etc. The more people; who can evacuate themselves from the area; the better. This will give the officers more time to focus on the people in need of assistance. To help identifying all the people in the evacuation area, all online mobile phones could be tracked, and marked as dots in an application. This function in itself could be an entire project.

## 6.7 Summary of Discussion

In previously chapter we have discussed the use a Geographical Information System (GIS) as a communication system, we have evaluated our project, and how it is usable in regards to Danish Emergency Rescue Services. We have also discussed our application, both in terms of verification and validation, and also in terms of a next generation, reflection and the source

which we have used. In the subsection ‘Our application’ we have discussed how much information can we include in our application without confusing and stressing the users, how many buttons we need in our system to operate with the maximum benefit, and including colours, size, fonts and naming, also whether the emergency commanders handle a pen while wearing his gloves in the midst of an emergency situation.

Afterwards we have discussed about GIS and the digital lifestyle and focus on the opportunities and needs for efficient solutions.

*The digital lifestyle puts a focus on both the opportunities and needs for efficient solutions that take advantage of the opportunities of the digital age. E-Government is a good example of how digital solutions can be used to streamline and optimize workflows. Sharing of data and use of information should be promoted and increasingly used as it will help to break down the silo approach and foster collaboration* (Hvingel & Hansen 2011).

So in this context we discuss what kind of geographic information and communications tool is important for the Danish Emergency Rescue Service and what about data quality and quantity. Another possibility for the next generation system, smartphones where it is possible for the public to take and submit GPS photo reports of disasters, which can be displayed on a map for others to view.

And finally in the source analysis we consider following points, credibility, up to date, bias and point of view and accuracy.

# 7 Conclusion

The aim of this project; as proclaimed in the introduction is to develop a prototype software system, which will support the communication between emergency response teams by the use of geographic information in an emergency situation, connected with catastrophes of flooding and saving lives and values.

During this project we have used Scrum as a project management method, we have adapted Scrum to fit our needs and the conditions which we work under. We have used open source software to develop our application. OpenLayers and PostGIS as our database, and GeoServer as our server application. We have taken considerations of security and maintenance of our database and server. We have decided to only use data hosted in Denmark, and preferably data secured by the responsibility of the Danish state; data where the state is responsible for quality, security, updating and accessibility.

Our prototypes are designed for use by DERS emergency commanders operating in the field and NOST users. Maybe someday it could be used as a source of inspiration to DERS, to develop their future geographic emergency rescue tool.

We have developed our application from scratch using HTML5, JavaScript and CSS. We have designed an intuitive and user friendly interface from a scientific point of view, with consideration to the end users' demands. After having tested our prototype we are satisfied with the result of our work. We have developed 2 online prototypes which we feel are suitable, pleasing and includes many functions. We believe that the prototypes can serve as a base towards a good and constructive dialogue with the DERS concerning their needs and possibilities.

From the beginning of this project we were ambitious as to the capabilities of our solution to be. We aimed for a comprehensive solution which unfortunately was not compatible with our resources and time perspective. This experience has given us an insight into the amount of unpredictable problems which can occur during software development.

We find that there are a few features which makes our software unique:

- This is a Danish system; developed in Denmark and include data which is stored in Denmark, any improvements can be performed in Denmark.
- We use data from data sets which covers all of Denmark; and which follow the INSPIRE and ISO standards.
- When our application is fully functional it will be possible to share and receive the common image of the situation, with all the actors at the site of an incident of flooding.
- Our application is adapted to the needs of the Danish Emergency Rescue Service's and the needs for field officers. We have designed two versions of our application adapted to their needs.
- The mobile solution which is adapted for a tablet, is designed for tablets with 10-inch screen and field use, with an intuitive user interface and large buttons.

Overall we are satisfied with our final software and the way we have worked as a group, we have learned a lot and have almost used of the knowledge which we have obtained during this MTM study program.

# Literature


- Bae & Hong (2012), 'Electrical properties of conductive fabrics for operating capacitive touch screen displays', *Textile Research Journal* March 2013 83: 329-336, first published on November 27, 2012, Sage Journals.
- Beaujardière (2002), *Web Map Service Implementation Specification*
- Beaujardière, 2006, Web Map Service Implementation Specification
- Beredskabsstyrelsen (2010), *Retningslinjer for indsatsledelse.*
- Blankenship, B. (2011), *Pro Agile.NET Development with Scrum*, ISBN-13 978-1-4302-3533-0
- Cai, G., Yu, B. and Chen, D. (2013), Modelling and Communicating the Conceptual Intent of Geo-Analytical Tasks for Human-GIS Interaction. *Transactions in GIS*, 17: 353–368. doi: 10.1111/tgis.12040
- Cockburn, A., 2007, 'Writing Effective Use Cases', The Agile Software Development Series, ISBN 0-201-70225-8
- Cosic, Z., Endersen, T., Foss, B. & Palmqvist, A. (2012), *Kommunikation med Geografisk Information i det danske beredskab* (*Communication With Geographic Information within the Danish Emergency Services*), Aalborg University, Institute of Planning
- Cosic, Z., Endersen, T., Foss, B. & Palmqvist, A. (2013), *Geografisk information som beredskabskommunikationsværktøj ved klimaskabteoversvømmelsesscenarier* (*Geographic Information as a Communications Tool in situations of Climate-made Flood Scenarios for the Danish Emergency Rescue Services*), Aalborg University, Institute of Planning
- Cronholm, S., Goldkuhl, G. & Ågerfalk, P. J. (1999), From Usability to Actability, *Computer and Information Science*, Linköping University Electronic Press, Vol. 4(1999): nr 5, ISSN 1401-9841
- Dueholm, K. & Laurentzius, M. (2002) *GPS*, ISBN 87-571-2412-4
- Engravslia, L., Gyland, L. F., Rafoss, T., Sletten, A. & Sælid, K. (2010), Open geospatial technology standards and their potential in plant pest risk management—GPS-enabled mobile phones utilising open geospatial technology standards Web Feature Service Transactions support the fighting of fire blight in Norway, Computers and Electronics in Agriculture, Volume 74, Issue 2.
- Fechner, T., Foerster, T., Fritze, H., Loock, F. & Remke, A. (2010), Low-cost satellite-based products for the Web – the example of Fire Web Service. In V. Bogorny & L. Vinhas (Eds.), Proceedings of 11th Brazilian Symposium on GeoInformatics. Presented

at the GeoInfo 2010, Sao Jose dos Campos: MCT/INPE., Institute for Geoinformatics (ifgi) - University of Muenster, Muenster, Germany. 52o North - Initiative for Geospatial Open Source software GmbH, Muenster, Germany

- Gade, S. (2005), Bekendtgørelse om risikobaseret kommunalt redningsberedskab, BEK nr 765 af 03/08/2005, Forsvarsministeriet, Forsvarsmin., j.nr. 04-1252-14
- GST (2013), *Project Proposal regarding Stakeholder involvement regarding national preparedness - Active environmental dialog*, Journal no. GST-736-00007.
- Hazzard E. (2011), *OpenLayers 2.10 Beginner's Guide*
- Holdener T.A. (2011), 'HTML5 Geolocation', *Bringing location to Web Application,* O'REILLY
- Herring, J.(2011), Open Geospatial Consortium Inc., OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture.
- Herring, J.(2010), Open Geospatial Consortium Inc., OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 2: SQL option
- Itkonen, J. & Rautiainen, R. (2005), Exploratory testing: A multiple case study. *In: Proceedings of the 4th International Symposium on Empirical Software Engineering (ISESE 2005)*. IEEE. ISBN 0-7803-9507-7
- Jihyun, B. & Kyung, H. H. (2012), 'Electrical properties of conductive fabrics for operating capacitive touch screen displays', *Textile Research Journal* March 2013 83: 329-336, first published on November 27, 2012
- Kohl, J. (2013), Demystifying Exploratory Testing, *Better software magazine - A Techwell Publication*, Volume 15, issue 5 - September/October 2013
- Krug, S. (2000), *Don't Make Me Think! A Common Sense Approach to Web Usability*, New Riders, ISBN-10: 0-321-34475-8, ISBN-13: 978-0-321-34475-5
- Krug, S. (2010), *Rocket Surgery Made Easy: The Do-It-Yourself Guide to Finding and Fixing Usability Problems*, New Riders, ISBN-13: 978-0-321-65729-9
- Ledsky, R. (2013), 33880 Geographical Information Systems (GIS) Tools: Fostering Healthcare, Health Communication Activities and Materials, *National Conference on Health Communication, Marketing, & Media,* 2013 Atlanta, GA, NPHIC: National Public Health Information Coalition
- Lacovella, S. & Youngblood, B. (2013), GeoServer; Beginner's Guide, Packt, ISBN 978-1-84951-668-6
- Lubbers P., Albers B. & Salim F. (2010), *Pro HTML5 Programming*.

- Lwin, K. K. & Murayama, Y. (2011), Web-Based GIS System for Real-Time Field Data Collection Using a Personal Mobile Phone, *Journal of Geographic Information System*, 2011, 3, 382-389, doi:10.4236/jgis.2011.34037
- Manolescu D., G. Santamaria (2002), *Java Thin Clients Revisited: An Architecture for Retrospective, Live Wireless Applications*
- Obe, R. O. & Hsu, L. S., (2011), PostGIS in action.
- OGC, (2006) Web Map Server Implementation Specification, ver. 1.3.0.
- OGC, (2007) Geography Markup Language (GML) Encoding Standard, ver. 3.2.1.
- OGC, (2010) Web Feature Service 2.0 Interface Standard, ver. 2.0.0
- OGC, (2011) Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture.
- Olsen, K. & Vinje, P. S. (2004), *Softwaretest – kom godt i gang* (*Software testing – get a good start*), Dansk IT – styrk dit netværk, ISBN 87-88972-34-8
- PCI Geomatics (2009), *GeoConference Client - User Guide*, Documentation version 3.0.1
- Peintner, B., Paolo Viappiani, N.S. (2008): Preferences in interactive systems: Technical challenges and case studies. *AI Magazine 29(4)*
- Perez A.S. (2012), *OpenLayers Cookbook*
- Pinde F., Sun J. (2010), Web GIS, Principles and applications
- Plewe, B. (1997), *GIS Online: Information, Retrieval, Mapping, and the Internet,* OnWord Press, ISBN 1-56690-137-5
- Pfost, D., Casady, W. & Shannon, K., (1998), Precision Agriculture: Global Positioning System (GPS), published by *University Extension*, University of Missouri-System
- Trevisani E. & Vitaletti A. (2004) Cell-ID location technique, limits and benefits: an experimental study
- Vretanos P. (2005), *Web Feature Service Implementation Specification*
- W.D. (2007), *Source Evaluation Guide*, Pulaski Technical College, Library
- Worboys, M. F. (1995), *GIS - A Computing Perspective*, Taylor & Francis, ISBN 0-7484-0064-8, ISBN 0-7484-0065-6

# Hyperlinks


- ArcGIS for Developers: https://developers.arcgis.com/en/javascript
- Bae & Hong, 'Electrical properties of conductive fabrics for operating capacitive touch screen displays', *Textile Research Journal*: http://trj.sagepub.com.zorac.aub.aau.dk/content/83/4/329.full.pdf+html
- Boundless, architecture: http://boundlessgeo.com/whitepaper/opengeo-architecture/
- CouchTools, for traction of movements: http://www.couchtools.com/
- Cronholm, S., Goldkuhl, G. & Ågerfalk, P. J. (1999), From Usability to Actability, Computer and Information Science, Linköping University Electronic Press: http://www.vits.org/publikationer/dokument/53.pdf
- Danish National Environmental Portal: http://internet.miljoeportal.dk/Sider/Forside.aspx
- Dixit, S. (2010), 'Running your web applications offline with HTML5 AppCache' http://dev.opera.com/articles/view/offline-applications-html5-appcache/
- DMI, Danish Meteorological Institute, severe weather warning in DK: http://www.dmi.dk/service-menu/varsel-beskrivelse/
- ECMA International: http://www.ecma-international.org/memento/index.html
- European GNSS Agency, applications: http://www.gsa.europa.eu
- FEMA (2013), Smartphone App: http://www.fema.gov/smartphone-app
- Geodata styrelsen, the Danish Geodata Agency: www.gst.dk
- Google Maps JavaScript API v3: https://developers.google.com/maps/documentation/javascript/?hl=fr-FR&csw=1
- GPS overview: http://www.csr.utexas.edu/texas_pwv/midterm/gabor/gps.html
- ISO, official home page: http://www.iso.org/iso/home.html
- ISO, ISO 19115:2003, Geographic information -- Metadata: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=26020
- ISO, ISO/TS 19139:2007, Geographic information -- Metadata -- XML schema implementation: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=32557
- ISO, ISO 19115-2:2009, Geographic information -- Metadata -- Part 2: Extensions for imagery and gridded data: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=39229

- ISO, ISO 19119:2005, Geographic information -- Services: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=39890
- ISO/TC211, geographic information/Geomatics: http://www.isotc211.org/
- Itkonen, J. & Rautiainen, R. (2005), Exploratory testing: A multiple case study. In: Proceedings of the 4th International Symposium on Empirical, Software Engineering: http://lib.tkk.fi/Diss/2011/isbn9789526043395/article2.pdf
- JavaScript library to transform point coordinates from one coordinate system to another: http://trac.osgeo.org/proj4js/
- jQuery JavaScript library: http://jquery.com
- Kortforsyningen: http://www.kortforsyningen.dk
- Mozilla Developer Network, MDN: https://developer.mozilla.org/en-US
- National Conference on Health Communication, Marketing, & Media (2013): https://cdc.confex.com/nphic/nphic13/webprogram/Paper33880.html
- Official U.S. Government information about the Global Positioning System (GPS) and related topics: http://www.gps.gov/systems/gps/space/
- OGC network tutorial: http://www.ogcnetwork.net/wfstutorial
- Open Geospatial official homepage /GML: http://www.opengeospatial.org/standards/gml
- Open Geospatial official homepage /WFS: http://www.opengeospatial.org/standards/wfs
- Open Geospatial official homepage /WMS: http://www.opengeospatial.org/standards/wms
- OpenLayers: http://openlayers.org
- Pfost, D., Casady, W. & Shannon, K. (1998), Precision Agriculture: Global Positioning System (GPS), University of Missouri Extension: http://extension.missouri.edu/explorepdf/envqual/wq0452.pdf
- Software Testing Help: www.softwaretestinghelp.com
- Touch screens gloves reviews: http://www.touchscreenglovereviews.com/
- W. D. (2007), Source Evaluation Checklist, Pulaski Technical College: http://www.pulaskitech.edu/library/content/source_evaluation_checklist.pdf
- World Wide Web Consortium, 2013 'Decomposition of HTML elements' http://www.w3.org

# Figures

# Tables

# Appendices

## Appendix 1 Guidelines for emergency rescue operations according to DEMA.

Priority tasks:

*Please notice that not all of the DEMA abbreviations in this appendix are rendered with their full name in this report.*

Police

- to determine the size of the emergency area and the focal area in cooperation with the other emergency commanders and the NOST
- to prepare and initiate the immediate evacuation
- to evaluate the need for warnings
- to coordinate the initiation of the necessary facilities according to the emergency situation: e.g. treatment centres, NOST, contact point, staging area, meeting points and temporary mortuaries
- to coordinate the efforts with the active units
- to plan and secure public access roads and routes for the emergency vehicles
- To give situation reports to the NOST
- provision of technical assistance
- to take the necessary precautions towards guarding and securing areas (inside and outside) this includes securing e.g. valuables
- to coordinate relations with the press within the focal area
- to consider the need for splitting the focal area in case of multiple sites of emergency situations
- to point out the command station (KST) officer in charge and radio-/log- operator
- To determine time for meeting/contact to ISL-R
- B, KOOL and other relevant officers in charge
- Evaluate the need for establishing an evacuation centre, in cooperation with the municipality crisis stab
- Evaluate the need for establishing an next of kin centre, in cooperation with the municipality crisis stab
- To coordinate logistics concerning barracks and lodging, catering, toilet facilities and
- evaluate in cooperation with ISL-RB – whether an area may be threatened (new danger area if the wind is turning)
- In addition, the police have a number of other police
- tasks, including: ensuring peace and order
- to initiate investigations to detect / identify survivors, injured or dead and to make documentation / log prospecting for use and evaluation

Emergency Management:

- To make situation assessments, including providing the necessary basis for a life-saving and damage remedial first aid

- to determine scope of a possible hazard area
- assessing the need for warning, see the warning Agreement, in consultation with ISL-PO
- to decide which first action to be taken
- inserting the team leaders for the attendance intervention units
- to determine - together with other efforts leadership - The location of the treatment area, command stage, contact point, moving area, marshalling centre and gathering place for the dead
- reconnaissance, including providing the necessary basis for the final emergency effort
- to initiate the final emergency effort
- to regularly assess whether the available resources at the site of injury is sufficient
- to initiate potential damage control of consequential damages
- to request the task-solving assistance required, as available assistance beyond its own units is requested through KST
- to provide arrival and situation reports to the control centre, to monitor and follow up the efforts undertaken with the aim of assessing the desired effect
- to establish the necessary technical management structure with the aim of creating time and space for damage location management
- to ensure that the emergency effort is carried out as efficiently as possible, and avoiding tying up unnecessary resources on the site of injury
- to continuously meet the inserted crew safety and to ensure clues and other evidence

### Defining the users and their needs

The guidelines from the Danish Emergency Management Agency (DEMA), states the different roles and their responsibility in response management. We have concentrated our project to have three different actors: the police in field, the emergency rescue in field and the coordination group.

### Phases for the emergency rescue

- situation assessment
- first efforts
- reconnaissance
- final efforts

### Situation assessment

At the emergency area, assessment is evaluated after:

- What can be seen
- Information from people
- People and animal in danger
- Defining the damaged area and area of danger?
- Categorising the damage
- Assessing the development in claims
- Special dangers
- Access roads

### First efforts

1. People rescue
2. Animal rescue
3. Salvage of values and the environment

4. Clearing of special dangers
5. Stopping the spread
6. Consideration of the final control of the damage

### Reconnaissance

In this stage the purpose for the emergency in charge is to determine the final efforts after:

- The extent of injury
- Damage development and special hazards
- Access roads

### Final efforts

The commander in charge gives the orders based on the reconnaissance. In of major emergency events the final efforts is divided into prioritised stages. The commander in charge has to continue the reconnaissance, control and follow up on the final efforts.

- to decide which first action to be taken
- inserting the team leaders for the attendance intervention units
- to determine - together with other efforts leadership - The location of the treatment area, command stage, contact point, moving area, marshalling centre and gathering place for the dead
- reconnaissance, including providing the necessary basis for the final emergency effort
- to initiate the final emergency effort
- to regularly assess whether the available resources at the site of injury is sufficient
- to initiate potential damage control of consequential damages

to request the task-solving assistance required, as available assistance beyond its own units is requested through KST to provide arrival and situation reports to the control centre

# Appendix 2 Use case examples on NOST users and evaluation

Evaluation of a flood and the rescue operation

**Primary actor:** The emergency commander

**Scope:** To Evaluation of emergency situations for future improvements.

**Level:** High level user goals.

**Stakeholders and interests:**
The emergency commander
NOST users

**Pre-condition:** Our application must be installed. High user authority is necessary.

**Minimal Guarantee:** Viewing registrations and information flow from an emergency situation.

**Success Guarantee:** In-depth evaluation of emergency situations, with clear indications of locations and time flow.

**Main Success Scenario:**
1. Open the geographic emergency tool on a pc
2. Go to the evaluation functionality
3. Select the registered emergency operation of interest
4. valuate

Guiding and aiding emergency commanders during flooding and rescue operations

**Primary actor:** NOST users

**Scope:** To guide and aid the emergency commanders in the field during flooding and emergency operations.

**Level:** High level user goals.

**Stakeholders and interests:**
The emergency commander
Emergency personnel
NOST users

**Pre-condition:** Our application must be installed. Online Access to databases is a necessity. Access to extensive amount of data and any GIS tools necessary

**Minimal Guarantee:** Identifying and following the locations of emergency commanders in the field during an emergency operation.

**Success Guarantee:** Identifying and following the locations of emergency commanders in

the field during an emergency operation. Supplying and aiding the emergency commanders in the field with any functionality and layer they ask for.

**Main Success Scenario:**

1. Open the geographic emergency tool on a pc
2. Create new emergency operation session for a flooding situation
3. Identifying and following the actors in the field
4. Communication with the actors in the field
5. Aiding and supporting the actors in the field with additional layers, and running functionalities
6. Keeping the full overview to coordinate the combined efforts of the emergency response teams.
7. Save and close the session for later evaluation

# Appendix 3 Extended version of our use case

Reporting and sharing the status of a flooding and the rescue operations

**Primary actor:** The emergency commander

**Scope:** Sharing the geographic 'image' of an emergency situation, drawing and sharing the focal area, drawing and sharing the location of road blocks, and finally drawing a polygon to select the number of residents within the geographic area.

**Level:** High level user goals.

**Stakeholders and interests:**
The emergency commander
Emergency Personnel
NOST users

**Pre-condition:** Our application must be installed. Online access to databases is also a necessity.

**Minimal Guarantee:** Drawing in the map and sharing the results

**Success Guarantee:** All parties involved in the operation can see the drawn objects in their geographical emergency tool and the changes has been logged with geographic references and a timestamp.

**Main Success Scenario:**

1. Open the geographic emergency tool
2. Map view identifies the primary actor's geographic location and zoom in automatically
3. Draw focal area
4. Place a road block
5. Identify the local residents affected by the situation

In connection with points 3-5 the information is shortly shared with the other users.

**Extensions:**

3a. Tab to mark the layer button 'Focal Area'
3b. Tab to mark the 'New' button
3c. By tabbing in the map view
    3c1. Tab to mark the vertices of the Focal Area polygon
    3c2. Double tab to close the polygon
3d. By drawing a polygon
    3d1. Place the 'finger' in the map view and draw a closed polygon
    3d2. Lifting the 'finger' closes the polygon
3e. No polygon is drawn
    3e1. An error message is given
    3e2. Returning to drawing a polygon
3f. Requirements to a drawn focal area are not meet

3f1. Edit Focal area
3f1a. Tab to mark the layer button 'Focal Area'
3f1b. Tab to mark the 'Edit' button
3f1c. Tab the polygon to be edited
3f1c1. Mark a vertex and drag it to the correct location
3f1c2. Tab a line between 2 vertices to add a new vertex
3f2. Delete polygon
3f2a. Tab to mark the layer button 'Focal Area'
3f1b. Tab the 'Delete' button
4a. Tab to mark the layer button 'Road Block'
4b. Tab to mark the 'New' button
4c. Tab in the map view to place the Road Block
4d. No point (road block) is drawn
4d1. An error message is given
4d2. Returning to drawing a polygon
4e. A road block does not meet the users requirements
4e1. Edit Roadblocks
4e1a. Tab to mark the layer button 'Road block'
4e1b. Tab to mark the 'Edit' button
4e1c. Tab the point to be edited
4e1d. Drag the point to the correct location
4e2. Delete Roadblock
4e2a. Tab to mark the layer button 'Road block'
4e2b. Tab the point to be edited
4e2c. Tab the 'Delete' button
5a. Tab the layer button 'Residents'
5b. Tab the button 'Select'
5c. By tabbing in the map view
5c1. Tab to mark the vertices of the selection area
5c2. Double tab to close the selection area
5d. By drawing a selection area
5d1. Place the 'finger' in the map view and draw a closed selection area
5d2. Lifting the 'finger' closes the selection area
5e. The selections area is no longer visible, but the residents within the selected area are highlighted, and a list of residents and their available information is formed.
5f. Nothing is selected
5f1. A message is returned that no residents are registered within the area
5f2. An error message is returned, the selection was not complete

6. Other users who are working at the same flooding incident are also represented by objects in the map view

7. Constantly following the changes of the operation reported by the other users

# Appendix 4 Requirements Outline

According to Alistair Cockburn (Cockburn, 2007):

*Chapter 1. Purpose and scope*
1a. What is the overall scope and goal?
1b Stakeholders (who cares?)
1c. What is in scope, what is out of scope?

*Chapter 2. Terms used / glossary*

*Chapter 3. The use cases*
3a. The primary actors and their general goals
3b. The business uses cases (operational concepts)
3c. The system use cases

*Chapter 4. The technology used*
4a. What technology requirements are there for this system?
4b. What systems will this system interface with, with what requirements?

*Chapter 5. Other requirements*
5a. Development process
- q1. Who are the project participants?
- q2. What values will be reflected (simple, soon, fast, or flexible)?
- q3. What feedback or project visibility do the users and sponsors want?
- q4. What can we buy, what must we build, what is our competition?
- q5. What other process requirements are there (testing, installation, etc.)?
- q6. What dependencies does the project operate under?

5b. Business rules
5c. Performance
5d. Operations, security, documentation
5e. Use and usability
5f. Maintenance and portability
5g. Unresolved and deferred

*Chapter 6. Human Backup, Legal, Political, Organizational Issues*
- q1. What is the human backup to system operation?
- q2. What legal and what political requirements are there?
- q3. What are the human consequences of completing this system?
- q4. What are the training requirements?
- q5. What assumptions, dependencies are there on the human environment?

# Appendix 5 Using WFS-T to modify data

One advantage of WFS over WMS is that you can do *transactions* to update the data set on the server directly. If you use OpenLayers, include the line in Listing 3[37] that creates a Save strategy. Use the OpenLayers DeleteFeature, DrawFeature, and ModifyFeature controls for quick, easy shape editing capabilities in your web application.

You might want to do edit operations directly from your own PHP code rather than try to edit the on-disk shapefile or PostGIS table directly. The API wrapper provides the execute WFS Transaction function, which accepts XML transaction statements. The namespace and layer are specified as part of the transaction.

Listing 4 illustrates running a simple WFS-T transaction that inserts an L-shaped line:

Listing 4. Running a WFS-T transaction to insert a small shape

```
$WFSTRequest = '
<?xml version="1.0" ?>
<wfs:Transaction service="WFS" version="1.1.0" xmlns:wfs="http://www.opengis.net/wfs"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/wfs
        http://schemas.opengis.net/wfs/1.1.0/wfs.xsd">
 <wfs:Insert>
   <feature:roads2 xmlns:feature="http://chrismichaelis.com/test_1">
     <feature:the_geom>
       <gml:MultiCurve srsName="EPSG:900913" xmlns:gml="http://www.opengis.net/gml">
         <gml:curveMember>
           <gml:LineString>
             <gml:posList>
               -11575958.058425 5537202.2479459 -11575346.562199
               5524666.5753089 -11561893.645222 5524666.5753089
             </gml:posList>
           </gml:LineString>
         </gml:curveMember>
       </gml:MultiCurve>
     </feature:the_geom>
   </feature:roads2>
 </wfs:Insert>
</wfs:Transaction>
';

$GeoServer->executeWFSTransaction($WFSTRequest);
```

In Listing 4[38], notice that coordinate projection matches what's stated in the srsName XML attribute.

Listing, creating, and deleting data

The sample PHP wrapper includes simple functions to list, delete, and create each of the major object types. When you create a PostGIS data store, the database must exist. When you use createLayer for a PostGIS table, the layer name is the table name. Make sure that the table contains at least one shape before you add the layer to GeoServer. When you use a Shapefile Directory data store, the layer name is the file name (without path), and the file

[37] http://www.ibm.com/developerworks/web/library/os-GeoServer/index.html#listing3
[38] http://www.ibm.com/developerworks/web/library/os-GeoServer/index.html#listing4

must exist in the data store location. Listing 1 illustrates the process of creating a workspace, data store, and layer in one quick operation:

Listing 1. Creating a workspace and PostGIS data store and publishing a layer

```
include "GeoServerWrapper.php";
$GeoServer = new GeoServerWrapper('http://geosrvr.chrismichaelis.com/GeoServer',
        $username, $password);

// public function createWorkspace($workspaceName)
$GeoServer->createWorkspace('delineation_34');

// public function createPostGISDataStore($datastoreName, $workspaceName,
//        $databaseName, $databaseUser, $databasePass, $databaseHost = 'localhost',
//        $databasePort = '5432')
$GeoServer->createPostGISDataStore('d34_outputs', 'delineation_34', 'delineations',
        'gisuser', 'gisuser', 'localhost');

// Create table
pg_query("CREATE TABLE d34_stream_output (...)");
// Add shape(s) according to your own program logic
pg_query("INSERT INTO d34_stream_output (...)");

// public function createLayer($layerName, $workspaceName, $datastoreName,
//        $description = '')
$GeoServer->createLayer('d34_stream_output', 'delineation_34', 'd34_outputs',
        'Delineation outputs for run #34');
```

Viewing layers

With the viewLayer function, you can request a data set in GML or in KML. Use the viewLayerLegend function to retrieve a legend image square for building your own legend layouts.

The PHP wrapper also provides a wfsPost function, which passes requests along to GeoServer. The sample tester.php API test class uses this function to capture and redirect requests from OpenLayers to GeoServer. Listing 2 shows the relevant code:

Listing 2. Capturing and redirecting GeoServer requests

```
// See if we're passing a WFS request to the server on private net
// (or for cross domain protection bypassing proxy)
$pathParts = explode('/', $_SERVER['PATH_INFO']);
if ($pathParts[1] == 'wfsauth') {
 $unused = array_shift($pathParts);
 $unused = array_shift($pathParts);
 $user = array_shift($pathParts);
 $pass = array_shift($pathParts);

 // Pass through for WFS using username and password to follow
 include "GeoServerWrapper.php";
 $GeoServer = new GeoServerWrapper('http://geosrvr.chrismichaelis.com/GeoServer',
        $user, $pass);
 $wfs = implode('/', $pathParts);
 if ($_SERVER['QUERY_STRING'] != '') $wfs .= '?' . $_SERVER['QUERY_STRING'];
 // public function wfsPost($apiPath, $post)
 echo $GeoServer->wfsPost($wfs, file_get_contents('php://input'));
 return;
} else if ($pathParts[1] == 'wfs') {
 $unused = array_shift($pathParts);
 $unused = array_shift($pathParts);
```

```
 // No auth required
 include "GeoServerWrapper.php";
 $GeoServer = new GeoServerWrapper('http://geosrvr.chrismichaelis.com/GeoServer');
 $wfs = implode('/', $pathParts);
 if ($_SERVER['QUERY_STRING'] != '') $wfs .= '?' . $_SERVER['QUERY_STRING'];
 echo $GeoServer->wfsPost($wfs, file_get_contents('php://input'));
 return;
}
```

Listing 3 shows how to connect OpenLayers to a WFS layer through this PHP proxy. Although the code and function name say wfs, the same approach works equally well if you prefer a WMS layer. WMS provides map imagery in Portable Network Graphics (PNG) format. WFS provide actual map data in GML format that is rendered on the client side by JavaScript.

Listing 3. Using the GeoServer WFS layer from OpenLayers

```
wfs = new OpenLayers.Layer.Vector("Editable Features", {
 strategies: [ new OpenLayers.Strategy.BBOX(), new OpenLayers.Strategy.Save() ],
     /* The preceding line enables WFS-T. */
 projection: new OpenLayers.Projection("EPSG:900913"),
     /* 900913 = Google projection, which is required to use Google Earth base layers */
 protocol: new OpenLayers.Protocol.WFS({
   version: "1.1.0",
   srsName: "EPSG:900913",
   url: "http://.../GeoServer_api/index.php/wfsauth/USERNAME/PASSWORD",
   featureNS :  "http://chrismichaelis.com/test_1",
   featurePrefix: 'test_1',
   featureType: "roads2",
   geometryName: 'the_geom',
   schema: "http://.../GeoServer_api/index.php/wfsauth/USERNAME/PASSWORD/" +
     "DescribeFeatureType?version=1.1.0&typename=test_1:roads2"
 })
});
map.addLayer(wfs);
```

You can also use this approach to implement some additional security. Suppose GeoServer is on a private network or behind a firewall. When you interact with it through your PHP application, check the request URL in detail to see which users of your web application have access to the requested layer.

# Appendix 6 Abstracts of ISO standards ISO 19115 and ISO 19119

## ISO 19115:2003, Geographic information -- Metadata[39]

ISO 19115:2003 defines the schema required for describing geographic information and services. It provides information about the identification, the extent, the quality, the spatial and temporal schema, spatial reference, and distribution of digital geographic data.

ISO 19115:2003 is applicable to:

- the cataloguing of datasets, clearinghouse activities, and the full description of datasets;
- geographic datasets, dataset series, and individual geographic features and feature properties.

ISO 19115:2003 defines:

- mandatory and conditional metadata sections, metadata entities, and metadata elements
- the minimum set of metadata required to serve the full range of metadata applications (data discovery, determining data fitness for use, data access, data transfer, and use of digital data);
- optional metadata elements - to allow for a more extensive standard description of geographic data, if required;
- a method for extending metadata to fit specialized needs.

Though ISO 19115:2003 is applicable to digital data, its principles can be extended into many other forms of geographic data such as maps, charts, and textual documents as well as non-geographic data.

NOTE Certain mandatory metadata elements may not apply to these other forms of data.

## ISO/TS 19139:2007, Geographic information -- Metadata -- XML schema implementation[40]

ISO/TS 19139:2007 defines Geographic MetaData XML (gmd) encoding, an XML Schema implementation derived from ISO 19115.

## ISO 19115-2:2009, Geographic information -- Metadata -- Part 2: Extensions for imagery and gridded data[41]

ISO 19115-2:2009 extends the existing geographic metadata standard by defining the schema required for describing imagery and gridded data. It provides information about the properties of the measuring equipment used to acquire the data, the geometry of the measuring process employed by the equipment, and the production process used to digitize the raw data. This extension deals with metadata needed to describe the derivation of geographic information from raw data, including the properties of the measuring system, and the numerical methods and computational procedures used in the derivation. The metadata required to address coverage data in general is addressed sufficiently in the general part of ISO 19115.

[39] http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=26020
[40] http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=32557
[41] http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=39229

### ISO 19119:2005, Geographic information -- Services[42]

ISO 19119:2005 identifies and defines the architecture patterns for service interfaces used for geographic information, defines its relationship to the Open Systems Environment model, and presents a geographic services taxonomy and a list of example geographic services placed in the services taxonomy. It also prescribes how to create a platform-neutral service specification, how to derive conformant platform-specific service specifications, and provides guidelines for the selection and specification of geographic services from both platform-neutral and platform-specific perspectives.

[42] http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=39890

# Appendix 7 How GPS works

GPS measurements are based on one way communication from satellites to a receiver on earth. By measuring the transmission time of a signal from satellite to the receiver, it is possible to measure the distance using the formula below:

> *s = c.t*
> *s: is the distance*
> *c: is the speed of the signal which is equal to the speed of light ( 3,0 108m/s)*
> *t: is the time which signal has been trough from satellite to receiver.*
>
> *(Dueholm, K., Laurentzius, M, 2002 'GPS', page 11)*

The satellite position is known for each time; therefore geographic coordinates for the receiver can be measured by a spatial binding. The principle is very simple; we can imagine having an object in a room which is bound by 3 ropes from the ceiling. If we know the lengths of the ropes, by stretching all three ropes at the same time, our object can be hanging only and only one place, see Figure 0-1.

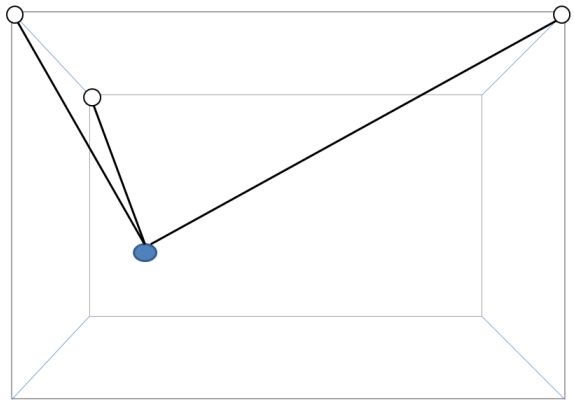


**Figure 0-1: spatial bounding shows how by knowing distances from three known points, make it possible to determine the coordinates for an unknown point (Dueholm and Laurentzius, 2002).**

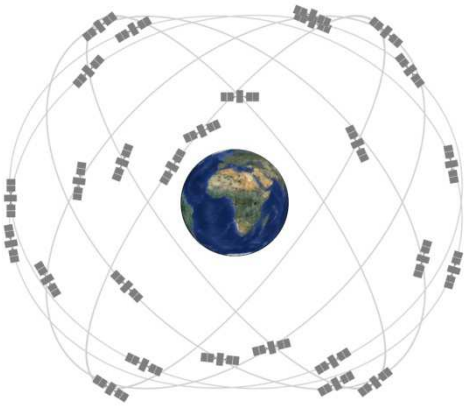


Figure 0-2: Shows the standard constellation of 24 GPS satellites (gps.gov)

NAVSTAR GPS consist of 24 satellites which are which flying in medium earth orbit of an altitude of approximately 20,200 km. Each satellite circulates the earth twice a day. To ensure having at least four satellites in view from any point on the earth, the GPS constellations[43] are arranged into six orbital planes surrounding the Earth with four satellites in each orbit, see Figure 0-2.

Exceeding the lifetime of the older generation of GPS satellites in one hand and demanding of modernization in another hand has made the US (United States) government replace the old GPS satellites with the new ones. The most modern operational GPS generation called GPS block Inline IF (IIF). In this new generation, besides longer life expectancy and higher accuracy, there is a third civil signal in a frequency protected, for the safety-of- life transportation. The newest and latest generation of GPS is called BLOCK III, which is under development; it is expected to provide more accurate and powerful signals.

[43] www.gps.gov/

# Appendix 8 List of pointers to keep in mind for a usability test

Here's a list of the things to keep in mind. If you read this list you'll be ready to start testing (Krug, 2000).

- **Try the test yourself first**. The day before the test, try doing whatever you're going to ask the test participants to do and make sure that you can do it in the time allotted. Make sure that whatever pages you're testing are accessible from the computer you'll be using, and that you have any passwords you'll need.
- **Protect the participants**. It's your responsibility to prevent any damage to your test users' self-esteem. Be nice to them. If they get stuck, don't let them get too frustrated. Be sure to pat them on the back (figuratively), and thank them (sincerely) when you're done. Let them know that their participation has been very helpful—exactly what you needed.
- **Be empathetic**. Be kind, patient, and reassuring. Make it clear to them that you know they're not stupid.
- **Try to see the thought balloons forming over their heads**. The main thing you're trying to do is observe their thought process. Whenever you're not sure what they're thinking, ask them. If the user has been staring at the screen for ten seconds, ask, 'What are you looking at?' or 'What are you thinking?'
  You're trying to see what their expectation is at each step and how close the site comes to matching that expectation. When they're ready to click, ask what they expect to see. After they click, ask if the result was what they expected.
- **Don't give them hints about what to do**. It's a lot like being a therapist. If they say, 'I'm not sure what to do next,' you should say, 'What do you think you should do?' or 'What would you do if you were at home?'
- **Keep your instructions simple**. There aren't very many, so you'll learn them quickly.
  - 'Look around the page and tell me what you think everything is and what you would be likely to click on.'
  - 'Tell me what you would click on next and what you expect you would see then.'
  - 'Try to think out loud as much as possible.'

  Don't be afraid to keep repeating them; it will be more boring to you than to the user.
- **Probe, probe, probe**. You have to walk a delicate line between distracting or influencing the users and finding out what they're really thinking, which they may not know themselves.
  For instance, when a user says, 'I like this page' you always want to ask a leading question like 'What do you like best about it?' If this produces 'Well, I like the layout' then you need to follow with 'What appeals to you about the layout?' You're looking for specifics, not because the specifics themselves are necessarily important but because eliciting them is the only way you can be sure you understand what the user is really reacting to.
- **Don't be afraid to improvise**. For instance, if the first two users get hopelessly stuck at the same point and it's obvious what the problem is and how to fix it, don't make the third user struggle with it needlessly. As soon as he encounters the problem, explain it and let him go on to something more productive.

- **Don't be disappointed if a user turns out to be inexperienced or completely befuddled**. You can often learn more by watching a user who doesn't get it than one who does. Because more experienced users have better coping strategies for 'muddling through,' you may not even notice that they don't get it.
- **Make some notes after each session**. Always take a few minutes right after each test session to jot down the main things that struck you. If you don't do it before you start the next test, it will be very hard to remember what they were.

  Being an observer at a usability test is a very cushy job. All you have to do is listen, watch closely, keep an open mind, and take notes.
- **Do they get it?** Without any help, can the users figure out what the site or the page is, what it does, and where to start?
- **Can they find their way around?** Do they notice and understand the site's navigation? Does your hierarchy—and the names you're using for things—make sense to them?
- **Head slappers.** You'll know these when you see them: the user will do something, or not do something, and suddenly everyone who's observing the session will slap his or her forehead and say, 'Why didn't we think of that?' or 'Why didn't we ever notice that?' These are very valuable insights.
- **Shocks.** These will also make you slap your head, but instead of saying 'Why didn't we notice that?' you'll say, 'How could she [the user] not notice that?' or 'How could she not understand that?' For instance, you might be shocked when someone doesn't notice that there is a menu bar at the top of each page, or doesn't recognize the name of one of your company's products.
  Unlike head slappers, the solution to shocks won't always be obvious and they may send you back to the drawing board.
- **Inspiration.** Users will often suggest a solution or the germ of a solution to a problem that you've struggled with for a long time. Very often the solution will be something you'd already thought about and rejected, but just watching someone actually encounter the problem will let you see it in a whole new light. And often something else about the project has changed in the meantime (you've decided to use a different technology, for instance, or there's been a shift in your business priorities) that makes an abandoned approach suddenly feasible.
- **Passion.** What are the elements of the site that users really connect with? Be careful not to mistake mere enthusiasm for passion, though. You're looking for phrases like 'This is exactly what I've been looking for!' or 'When can I start using this?'

  Guidelines to what to fix and what not to
- **Ignore 'kayak' problems.** In any test, you're likely to see several cases where users will go astray momentarily but manage to get back on track almost immediately without any help. It's kind of like rolling over in a kayak; as long as the kayak rights itself quickly enough, it's all part of the so-called fun. In basketball terms, no harm, no foul.

  As long as (a) everyone who has the problem notices that they're no longer headed in the right direction quickly, and (b) they manage to recover without help, and (c) it doesn't seem to faze them, you can ignore the problem. In general, if the user's second guess about where to find things is always right, that's good enough.

Of course, if there's an easy and obvious fix that won't break anything else, then by all means fix it. But kayak problems usually don't come as a surprise to the development team. They're usually there because of some ambiguity for which there is no simple resolution. For example, there are usually at least one or two oddball items that don't fit perfectly into any of the top-level categories of a site. So half the users may look for movie listings in Lifestyles first, and the other half will look for them in Arts first. Whatever you do, half of them are going to be wrong on their first guess, but everyone will get it on their second guess, which is fine.

- **Resist the impulse to add things.** When it's obvious in testing that users aren't getting something, most people's first reaction is to add something, like an explanation or some instructions.
  Very often, the right solution is to take something (or things) away that are obscuring the meaning, rather than adding yet another distraction.
- **Take 'new feature' requests with a grain of salt.** People will often say, 'I'd like it better if it could do x.' It always pays to be suspicious of these requests for new features. If you probe deeper, it often turns out that they already have a perfectly fine source for x and wouldn't be likely to switch; they're just telling you what they like.
- **Grab the low-hanging fruit.** The main thing you're looking for in each round of testing is the big, cheap wins. These fall into two categories:
  - Head slappers. These are the surprises that show up during testing where the problem and the solution were obvious to everyone the moment they saw the first user try to muddle through. These are like found money, and you should fix them right away.
  - Cheap hits. Also try to implement any changes that (a) require almost no effort, or (b) require a little effort but are highly visible.
- **Don't Throw the Baby Out with the Dishes.** Like any good design, successful Web pages are usually a delicate balance, and it's important to keep in mind that even a minor change can have a major impact. Sometimes the real challenge isn't fixing the problems you find—it's fixing them without breaking the parts that already work.

  Whenever you're making a change, think carefully about what else is going to be affected. In particular, when you're making something more prominent than it was, consider what else might end up being de-emphasized as a result.

# Appendix 9 Checklist for NON-functional testing

Validation of the user interface

- Spelling
- All functions present
- All links functional and correct
- Order of buttons and functions
- Colours, fonts, sizes, etc.
- The active buttons are identifiable
- Map view is in the right location

# Appendix 10 Exploratory testing

## Some Exploratory Testing Styles (Kohl, 2013)

**Intuitive:** A common style, tend to come naturally to testes that have not learned specific exploratory testing techniques.

**Learning Dominant:** While working to get to know the software, the testers both learn about the software and find bugs. Learning about the application leads to bug discoveries and test ideas.

**Systematic:** Advanced exploratory testing involves using a particular system, model, or strategy to guide the testing. Use of abstract models during test execution to guide and help the exploratory testing be more repeatable and consistent.

**Regression:** Exploratory testers often use lightweight coverage outlines as guidance rather than test cases. The guidelines help ensure easy and frequent repeated testing in desired areas.

## Exploratory Test Guidance (Kohl, 2013)

**Bug report:** List for bug fix validations

**Test ideas**

**Test goals**

**Charter for a testing session**

**Test checklists**

**Coverage outlines**

**Visual coverage**

**Feature maps**

**Test Cases**

# Appendix 11 WFS, GetCapabilities

This XML file does not appear to have any style information associated with it. The document tree is shown below.


<wfs:WFS_Capabilities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://www.opengis.net/wfs" xmlns:wfs="http://www.opengis.net/wfs" xmlns:ows="http://www.opengis.net/ows" xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc" xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:Test="http://opengeo.org/#Test" xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst" xmlns:medford="http://medford.opengeo.org" xmlns:MTM="MTM_data" xmlns:opengeo="http://opengeo.org" xmlns:usa="http://usa.opengeo.org" xmlns:world="http://world.opengeo.org" version="1.1.0" xsi:schemaLocation="http://www.opengis.net/wfs http://mtm-gruppe4.no-ip.org:8080/geoserver/schemas/wfs/1.1.0/wfs.xsd" updateSequence="458">
<ows:ServiceIdentification>
<ows:Title>GeoServer Web Feature Service</ows:Title>
<ows:Abstract>
This is the reference implementation of WFS 1.0.0 and WFS 1.1.0, supports all WFS operations including Transaction.
</ows:Abstract>
<ows:Keywords>
<ows:Keyword>WFS</ows:Keyword>
<ows:Keyword>WMS</ows:Keyword>
<ows:Keyword>GEOSERVER</ows:Keyword>
</ows:Keywords>
<ows:ServiceType>WFS</ows:ServiceType>
<ows:ServiceTypeVersion>1.1.0</ows:ServiceTypeVersion>
<ows:Fees>NONE</ows:Fees>
<ows:AccessConstraints>NONE</ows:AccessConstraints>
</ows:ServiceIdentification>
<ows:ServiceProvider>
<ows:ProviderName/>
<ows:ServiceContact>
<ows:IndividualName>MTM-Gruppe4</ows:IndividualName>
<ows:PositionName>Student</ows:PositionName>
<ows:ContactInfo>
<ows:Phone>
<ows:Voice>0045123456</ows:Voice>
<ows:Facsimile/>
</ows:Phone>
<ows:Address>
<ows:City>Denmark</ows:City>
<ows:AdministrativeArea/>
<ows:PostalCode/>
<ows:Country>Danmark</ows:Country>
</ows:Address>
</ows:ContactInfo>
</ows:ServiceContact>
</ows:ServiceProvider>
<ows:OperationsMetadata>
<ows:Operation name="GetCapabilities">
<ows:DCP>
<ows:HTTP>
<ows:Get xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
<ows:Post xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
</ows:HTTP>
</ows:DCP>
<ows:Parameter name="AcceptVersions">
<ows:Value>1.0.0</ows:Value>
<ows:Value>1.1.0</ows:Value>
</ows:Parameter>
<ows:Parameter name="AcceptFormats">
<ows:Value>text/xml</ows:Value>
</ows:Parameter>
</ows:Operation>
<ows:Operation name="DescribeFeatureType">
<ows:DCP>
<ows:HTTP>
<ows:Get xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
<ows:Post xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
</ows:HTTP>

```
</ows:DCP>
<ows:Parameter name="outputFormat">
<ows:Value>text/xml; subtype=gml/3.1.1</ows:Value>
</ows:Parameter>
</ows:Operation>
<ows:Operation name="GetFeature">
<ows:DCP>
<ows:HTTP>
<ows:Get xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
<ows:Post xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
</ows:HTTP>
</ows:DCP>
<ows:Parameter name="resultType">
<ows:Value>results</ows:Value>
<ows:Value>hits</ows:Value>
</ows:Parameter>
<ows:Parameter name="outputFormat">
<ows:Value>text/xml; subtype=gml/3.1.1</ows:Value>
<ows:Value>GML2</ows:Value>
<ows:Value>GML2-GZIP</ows:Value>
<ows:Value>SHAPE-ZIP</ows:Value>
<ows:Value>application/gml+xml; version=3.2</ows:Value>
<ows:Value>application/json</ows:Value>
<ows:Value>csv</ows:Value>
<ows:Value>gml3</ows:Value>
<ows:Value>gml32</ows:Value>
<ows:Value>json</ows:Value>
<ows:Value>text/xml; subtype=gml/2.1.2</ows:Value>
<ows:Value>text/xml; subtype=gml/3.2</ows:Value>
</ows:Parameter>
<ows:Constraint name="LocalTraverseXLinkScope">
<ows:Value>2</ows:Value>
</ows:Constraint>
</ows:Operation>
<ows:Operation name="GetGmlObject">
<ows:DCP>
<ows:HTTP>
<ows:Get xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
<ows:Post xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
</ows:HTTP>
</ows:DCP>
</ows:Operation>
<ows:Operation name="LockFeature">
<ows:DCP>
<ows:HTTP>
<ows:Get xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
<ows:Post xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
</ows:HTTP>
</ows:DCP>
<ows:Parameter name="releaseAction">
<ows:Value>ALL</ows:Value>
<ows:Value>SOME</ows:Value>
</ows:Parameter>
</ows:Operation>
<ows:Operation name="GetFeatureWithLock">
<ows:DCP>
<ows:HTTP>
<ows:Get xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
<ows:Post xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
</ows:HTTP>
</ows:DCP>
<ows:Parameter name="resultType">
<ows:Value>results</ows:Value>
<ows:Value>hits</ows:Value>
</ows:Parameter>
<ows:Parameter name="outputFormat">
<ows:Value>text/xml; subtype=gml/3.1.1</ows:Value>
```

<ows:Value>GML2</ows:Value>
<ows:Value>GML2-GZIP</ows:Value>
<ows:Value>SHAPE-ZIP</ows:Value>
<ows:Value>application/gml+xml; version=3.2</ows:Value>
<ows:Value>application/json</ows:Value>
<ows:Value>csv</ows:Value>
<ows:Value>gml3</ows:Value>
<ows:Value>gml32</ows:Value>
<ows:Value>json</ows:Value>
<ows:Value>text/xml; subtype=gml/2.1.2</ows:Value>
<ows:Value>text/xml; subtype=gml/3.2</ows:Value>
</ows:Parameter>
</ows:Operation>
<ows:Operation name="Transaction">
<ows:DCP>
<ows:HTTP>
<ows:Get xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
<ows:Post xlink:href="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfs"/>
</ows:HTTP>
</ows:DCP>
<ows:Parameter name="inputFormat">
<ows:Value>text/xml; subtype=gml/3.1.1</ows:Value>
</ows:Parameter>
<ows:Parameter name="idgen">
<ows:Value>GenerateNew</ows:Value>
<ows:Value>UseExisting</ows:Value>
<ows:Value>ReplaceDuplicate</ows:Value>
</ows:Parameter>
<ows:Parameter name="releaseAction">
<ows:Value>ALL</ows:Value>
<ows:Value>SOME</ows:Value>
</ows:Parameter>
</ows:Operation>
</ows:OperationsMetadata>
<FeatureTypeList>
<Operations>
<Operation>Query</Operation>
<Operation>Insert</Operation>
<Operation>Update</Operation>
<Operation>Delete</Operation>
<Operation>Lock</Operation>
</Operations>
<FeatureType xmlns:MTM="MTM_data">
<Name>MTM:hoejbebyg_po</Name>
<Title>Hoej bebyggele i project område</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>features</ows:Keyword>
<ows:Keyword>hoejbebyg_po</ows:Keyword>
<ows:Keyword>high buildings</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.252697461032465 55.55303406957044</ows:LowerCorner>
<ows:UpperCorner>12.695031134983887 55.87068559083786</ows:UpperCorner>
</ows:WGS84BoundingBox>
<FeatureType xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst">
<Name>wfst:adressermpo</Name>
<Title>adressermpo</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>adressermpo</ows:Keyword>
<ows:Keyword>features</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.486090901158793 55.6397502772762</ows:LowerCorner>

<ows:UpperCorner>12.585177712844912 55.702167619556356</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst">
<Name>wfst:adresserpomini</Name>
<Title>adresserpomini</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>features</ows:Keyword>
<ows:Keyword>Adresser</ows:Keyword>
<ows:Keyword>Address</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.160045611365963 55.44628118259497</ows:LowerCorner>
<ows:UpperCorner>12.707285804757152 55.8978689924265</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst">
<Name>wfst:bykerne1_po</Name>
<Title>bykerne1_po</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>bykerne1_po</ows:Keyword>
<ows:Keyword>features</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.476220707462264 55.62760193880749</ows:LowerCorner>
<ows:UpperCorner>12.633814462586574 55.7565115047244</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst">
<Name>wfst:bykerne_po</Name>
<Title>bykerne_po_wfst</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>bykerne_po</ows:Keyword>
<ows:Keyword>features</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.476220707462264 55.62760193880749</ows:LowerCorner>
<ows:UpperCorner>12.633814462586574 55.7565115047244</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst">
<Name>wfst:bykernempo</Name>
<Title>bykernempo</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>bykernempo</ows:Keyword>
<ows:Keyword>features</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.48534122728938 55.64528602776348</ows:LowerCorner>
<ows:UpperCorner>12.587995803399695 55.703723407676975</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:MTM="MTM_data">
<Name>MTM:industri_po</Name>
<Title>industri i Project område</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>features</ows:Keyword>
<ows:Keyword>industri_po</ows:Keyword>

```
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.154446271275177 55.44358714010092</ows:LowerCorner>
<ows:UpperCorner>12.701911795298566 55.90058465735619</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:opengeo="http://opengeo.org">
<Name>opengeo:lavbebyg_po</Name>
<Title>lavbebyg_po</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>lavbebyg_po</ows:Keyword>
<ows:Keyword>features</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.14393790933558 55.44174822693043</ows:LowerCorner>
<ows:UpperCorner>12.813167532072418 55.909582904519695</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:MTM="MTM_data">
<Name>MTM:lavbebyg_po</Name>
<Title>lavbebygggelse i Projekt område</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>lavbebyg_po</ows:Keyword>
<ows:Keyword>features</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.14393790933558 55.44174822693043</ows:LowerCorner>
<ows:UpperCorner>12.813167532072418 55.909582904519695</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst">
<Name>wfst:lavbebygmpo</Name>
<Title>lavbebygmpo</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>lavbebygmpo</ows:Keyword>
<ows:Keyword>features</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.4772795174601 55.63502605616832</ows:LowerCorner>
<ows:UpperCorner>12.58779504656476 55.70315564306548</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst">
<Name>wfst:lavningersydhavn</Name>
<Title>lavningersydhavn</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>features</ows:Keyword>
<ows:Keyword>lavningersydhavn</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.528511911587778 55.64621645642365</ows:LowerCorner>
<ows:UpperCorner>12.558119274006827 55.657895556976236</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst">
<Name>wfst:lavningersydhavn5m</Name>
<Title>lavningersydhavn5m</Title>
<Abstract/>
```

<ows:Keywords>
<ows:Keyword>features</ows:Keyword>
<ows:Keyword>lavningersydhavn5m</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.528428578627183 55.64616931444639</ows:LowerCorner>
<ows:UpperCorner>12.558202660223754 55.65794268261602</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst">
<Name>wfst:test</Name>
<Title>test</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>test</ows:Keyword>
<ows:Keyword>features</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.577054188152704 55.66183348372526</ows:LowerCorner>
<ows:UpperCorner>12.5825369983428 55.66448913165408</ows:UpperCorner>
</ows:WGS84BoundingBox>
</FeatureType>
<FeatureType xmlns:wfst="http://mtm-gruppe4.no-ip.org:8080/geoserver/wfst">
<Name>wfst:vej</Name>
<Title>vej</Title>
<Abstract/>
<ows:Keywords>
<ows:Keyword>vej</ows:Keyword>
<ows:Keyword>features</ows:Keyword>
</ows:Keywords>
<DefaultSRS>urn:x-ogc:def:crs:EPSG:25832</DefaultSRS>
<ows:WGS84BoundingBox>
<ows:LowerCorner>12.37433041013701 55.5947544939768</ows:LowerCorner>
<ows:UpperCorner>12.738563761953193 55.715727538958376</ows:UpperCorner>
</ows:WGS84BoundingBox>

# Appendix 12 Prototype versions

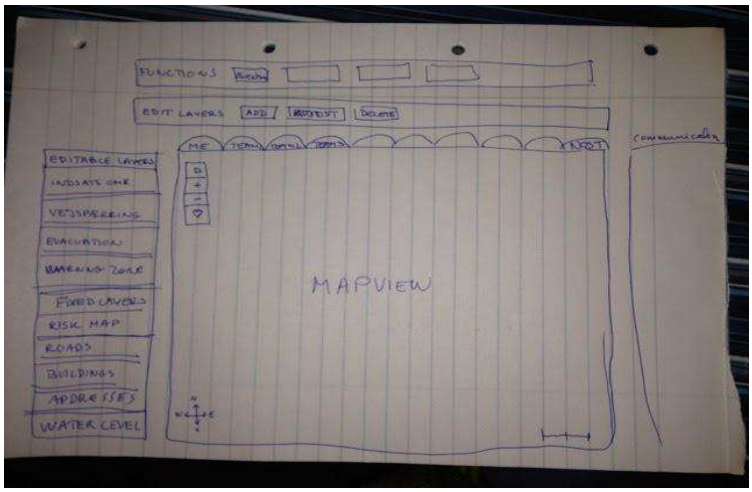



Figure 0-3: Prototype version 1

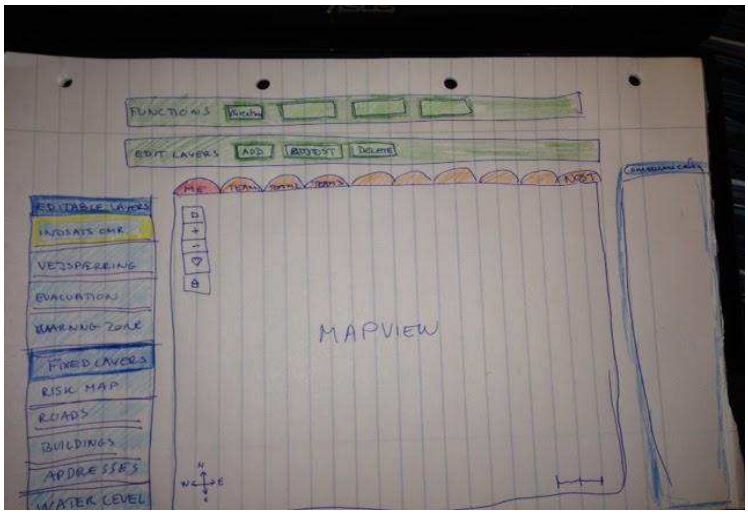



Figure 0-4: Prototype version 2

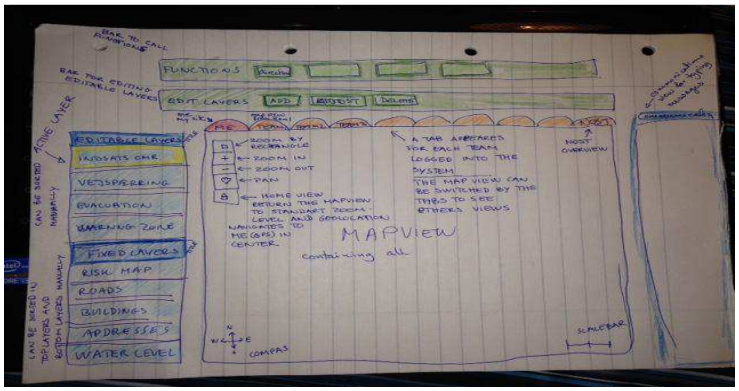



Figure 0-5: Prototype version 3

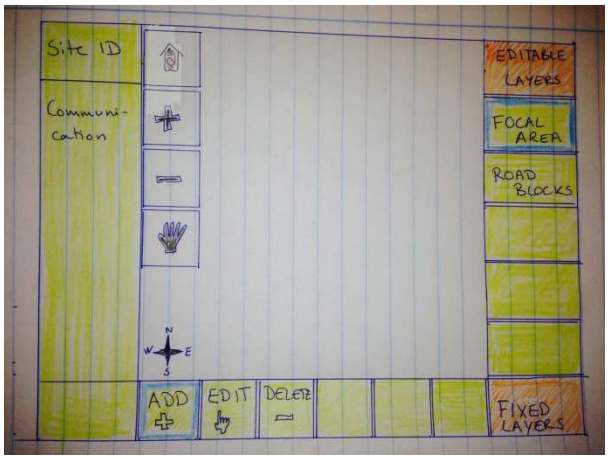



Figure 0-6: Prototype version 4

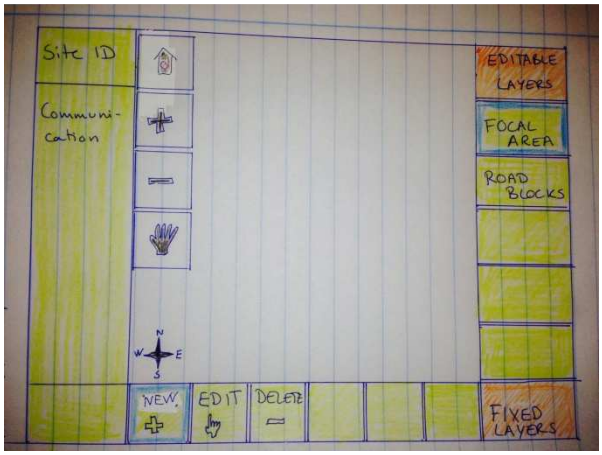



Figure 0-7: Prototype version 5

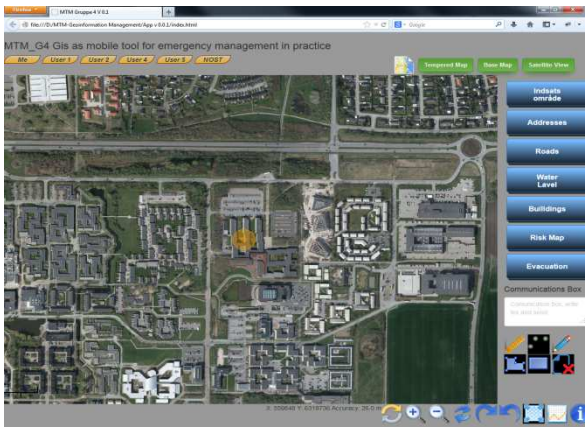



Figure 0-8: Prototype version 6

# Appendix 13 Error Description Table

Table of error found during our build-up and implementation of our software:

| Tester | Test ID | Error description | Error location | Description of the correction | Problem solver |
|---|---|---|---|---|---|
| Tina | T001 | Zooming fails | Map tools | | Zijad |
| Bjarke | B001 | address loading from AWS.dk | csv load failing, missing directory rights. | Excel + loading from QGIS | Bjarke |
| Bjarke | B002 | spatial_ref_sys table in pgadmin from database is missing | https://vilimpoc.org/blog/2012/12/04/spatial_ref_sys-and-geometry_columns/ | Create table by script + load data by running spatial_ref_sys.sql* in SQL pane. *c:programmer(x86)/OpenGeo/OpenGeo Suite/pgsl/9.1/share/contrib/postgis-2.0 | Bjarke |
| Azad | A001 | port 8080 in ruter can't open for GeoServer | Router setting | Port 8080 was dedicated to internet supplier for support | Azad |
| Zijad | Z001 | Selection button dont work | App | Adjusting the JavaScript code | Azad |
| Azad | A002 | Simple polygon is uploaded as multipolygon in database | Database | changing the options in database | Azad |
| Azad | A002 | GeoServer cannot start | Geoserver | Lack of a file in installation | Azad |
| Bjarke | B003 | test service ports closed | Router | Bridge mode (provider) + new router | Bjarke |
| Bjarke | B004 | TomCat/GeoServer WAR | WAR size | OpenGeo | Group |
| Bjarke | B005 | DHCP dynamic ip address | Provider | MAC address to server ip | External |
| | | | | | |