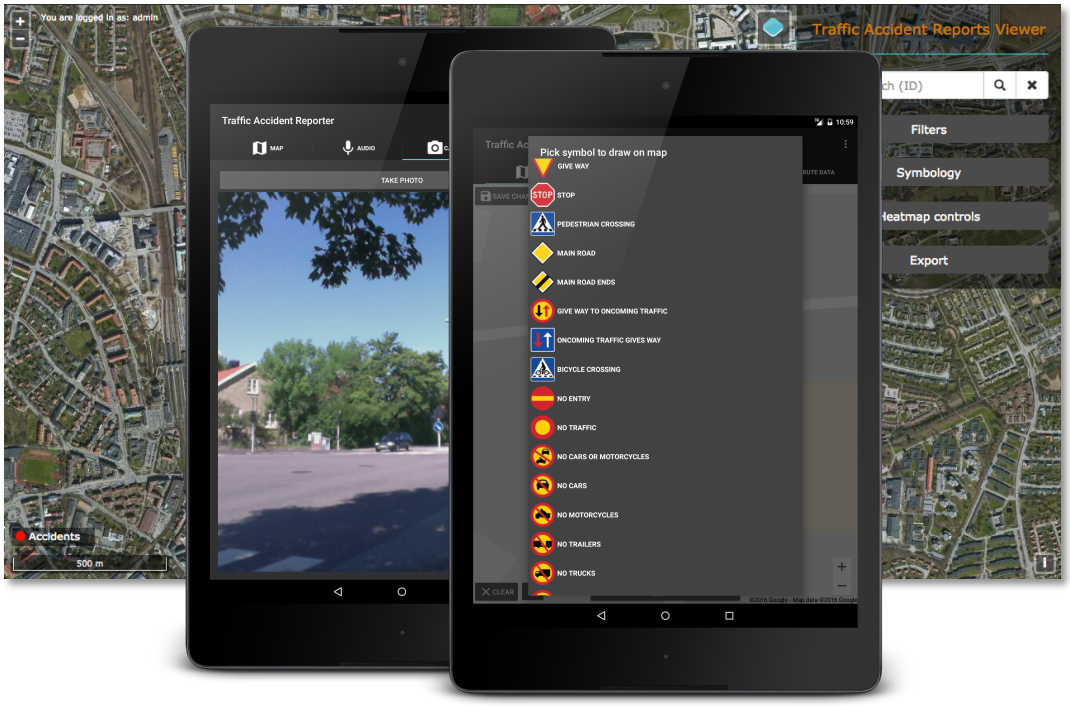
Student thesis series INES nr 403

Design and development of a prototype mobile geographic information system for real-time collection and storage of traffic accident data



Peter Markus

2016
Department of
Physical Geography and Ecosystem Science
Lund University
Sölvegatan 12
S-223 62 Lund
Sweden

Peter Markus (2016).

Design and development of a prototype mobile geographic information system for real-time collection and storage of traffic accident data (English)
Utveckling av prototyp för mobilt geografiskt informationssystem för insamling och lagring av trafikolycksdata i realtid (Swedish)
Master degree thesis, 30 credits in *Geomatics*
Department of Physical Geography and Ecosystem Science, Lund University

Level: Master of Science (MSc)

Disclaimer

This document describes work undertaken as part of a program of study at the University of Lund. All views and opinions expressed herein remain the sole responsibility of the author, and do not necessarily represent those of the institute.

Design and development of a prototype mobile geographic information system for real-time collection and storage of traffic accident data

---

Peter Markus

Master thesis, 30 credits, in Geomatics

Ali Mansourian
Supervisor

Roger Groth
Supervisor

Exam committee:
Examiner 1, Petter Pilesjö
Examiner 2, Lars Harrie

Acknowledgements

Extra gratitude is extended to Ali Mansourian for his ability to always be available to provide feedback when supervision was needed, and for his ability to always stay positive. Thank you Ali.

Abstract

Today, Swedish police authorities nationwide collect data of traffic accidents. The information is stored in a national database managed by the Swedish Transport Agency and is an important resource in the process of analysing and improving road safety.

Literature studies in this thesis, together with earlier work by the author have suggested that the data collection process is in need of an update: a digital tool for such data collection is necessary. Problems with varying quality of data and long submission times of reports have been attributed to the current method, which involves a paper form. A digital method including a handheld device is expected to improve data quality and shorten overall submission times.

The aim of the thesis has thus been to design and develop a mobile GIS system for collection and management of traffic accident information for police authorities.

The project has utilized mainly open source tools. The result is a system containing an Android application for data collection, a database server with a database for storage, an application server/web server to host a software server (map server), and a web server that handles requests and hosts a web service for viewing and retrieving data.

The created system can collect all of the information that the currently used analogue method does as well as new media such as GPS coordinates, photographs and audio. The functionality of the web service demonstrates that data is collected and stored in suitable formats in a database schema that is flexible enough to facilitate a wide range of queries relevant to the field of road safety.

Key words: STRADA, Open Source, Web-GIS, Android, WFS, GeoServer, OpenLayers, Web service, PostgreSQL, PostGIS, Tomcat, Apache, PHP, LBS.

List of abbreviations

STRADA – Swedish TRaffic Accident Data Acquisition

OGC – Open Geospatial Consortium

WMS – Web Map Service

WFS – Web Feature Service

GML – Geography Markup Language

FOSS – Free and Open Source Software

API – Application Programming Interface

LBS – Location-Based Service

CSS – Cascading Style Sheets

SLD – Styled Layer Descriptor

Table of Contents

1 Introduction

1.1 Problem definition

In 2012 the author conducted a quantitative analysis of bicycle related traffic accidents in the municipality of Malmö. Data was provided as an Excel spreadsheet by the municipality and consisted of accidents that occurred during the period 2005-2010. The data was produced by the Swedish Transport Agency for STRADA.

During analysis of the data, questions emerged regarding the quality and usefulness of the data. The report already considered the problem that not all accidents where people are injured are reported. However, during analysis of the geographical distribution of accidents based on light conditions in which they occurred it was noted that 349 of the 2942 events were missing coordinates and that 989, i.e. more than a third of the events, were missing information about light conditions in the attribute data (Markus 2012). By deciding to analyse two attributes, light conditions and location, more than a third of the data could not be used. The amount of useful data would decrease as more attributes were used in analysis. With such a decrease in population one might have to question the generalizability of the end results.

The level of detail of attribute data varied between the reported accidents. Of those reports that did have coordinates, the geographic precision appeared to vary. The varying level of detail in attribute data made it necessary to visit the locations to deduce the possible causes of the accident. The study left me with the impression that there must be potential to improve the quality and quantity of the data by improving the production process. The data collection process had been carried out by writing reports on paper. Changing the medium to a digital format should be a potential solution to obtain a simpler and more structured data collection process resulting in higher quality of data. A suitable medium instead of paper could be a handheld computer, or tablet.

1.2 Background - A future without fatalities on Swedish roads

In 1993 the Swedish transport agency was given the task by the parliament to investigate how statistics of traffic accidents could improve preventive traffic safety work (Sjöö and Ungerbäck 2007). The result was a report published in 1996 containing the proposition for a system used nationwide by all agencies involved in traffic safety (Vägverket 1996).

According to the report, the system would provide a more realistic presentation of the safety of Swedish roads. All future traffic victims that would receive medical treatment were going to be part of the new statistics (Vägverket 1996). The report argued that the new data would provide a better basis for priorities and decision making leading to more efficient safety work, i.e. a decreasing number of yearly victims (Vägverket 1996).

Later in 1996, the parliament gave the Swedish Transport Agency the task of creating and launching the proposed system (Swedish Transport Agency n.d.-a). According to the instructions, the three main goals for the new system were to support traffic safety work on central, regional, and local levels, provide basis for correct decisions regarding new safety measures, and minimize costs and duplication of work within public administration (Sjöö and Ungerbäck 2007). In 1997 the Swedish parliament approved a proposition regarding new nationwide strategies for traffic safety work (Sjöö and Ungerbäck 2007). The long-term objective was to reduce the amount of yearly fatalities and serious injuries in road related accidents to zero (Persson and Uusmann 1997). Development and initialization of the new system was started between 1998 and 2000. During this period the project consisted of seven subgroups based on geographical administrative boundaries. Each subgroup incorporated representatives from police, medical services, and road authorities (Swedish Transport Agency n.d.-a). The system that was implemented is named STRADA (see 1.2.1 STRADA).

1.2.1 STRADA

STRADA, or Swedish TRaffic Accident Data Acquisition (Sjöö and Ungerbäck 2007), is a system for storing, managing, and distributing nationwide traffic accident data reported by police supplemented by reports from hospitals (Swedish Transport Agency n.d.-c). The system is now part of the Swedish Transport Agency. Police are providing data nationwide for STRADA since the end of 2002 (Sjöö and Ungerbäck 2007). More than half of the municipalities of Sweden have access to STRADA and use the data for analysis before carrying out road safety operations. Other users of the data are e.g. The Swedish Transport Agency, rescue services and police (Swedish Transport Agency n.d.-b).

1.2.2 Reporting to STRADA

The first step of reporting an accident to STRADA (see Figure 1.1) occurs when a police officer, with a pen, fills out a paper form (see Appendix A) describing the accident and the surrounding conditions (Mattsson and Ungerbäck 2013). This should be done in situ and as soon as possible (1§ SFS 1965:561). The process includes the drawing of a reproduction of the scene using specific symbology suitable for handwriting (Mattsson and Ungerbäck 2013). As guidance, police authorities have access to documentation in the form of a tutorial on how to complete the form (Sjöö and Ungerbäck 2007; Mattsson and Ungerbäck 2013). Upon completion, the form should be checked by the police officer in charge before it is forwarded to staff within the same precinct who are responsible for registering the data in STRADA (Mattsson and Ungerbäck 2013).

At the second step, the data is digitized and submitted to STRADA databases located in the city of Borlänge using a client called the police web (see Figure 1.1). The software also encrypts personal information before submission (Sjöö and Ungerbäck 2007).

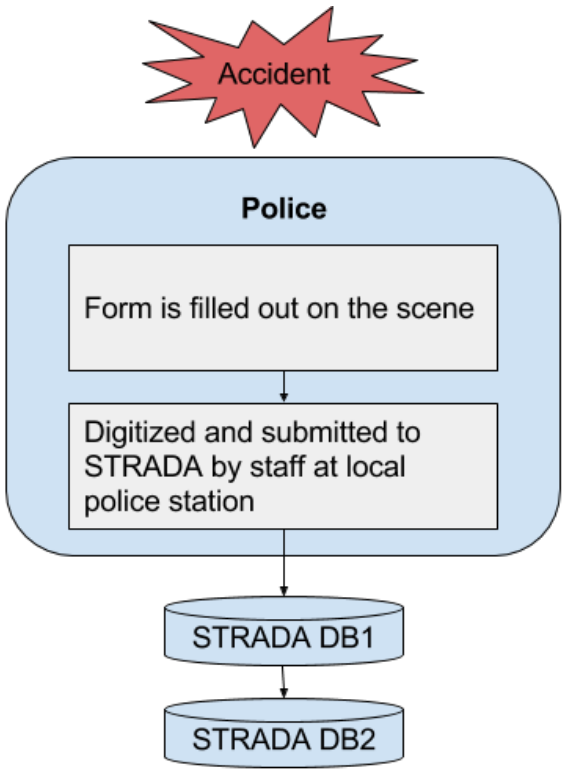


Figure 1.1. The process from accident to data ready for use. Database number 1 stores all data including personal information, however the personal information is encrypted. Database number 2 contains no personal information. This is the database that is shared and used to produce statistics. Based on the original figure by Sjöö & Ungerbäck (2007).

Accidents that have resulted in personal injuries should be reported to the Swedish Transport Agency as soon as possible and the latest seven days after police authorities have been made aware of them (1§ SFS 1965:561). Fatal accidents should also be reported as soon as possible but the latest five days after police authorities have been made aware of them (2§ SFS 1965:561).

Users access data in STRADA database number two through a specific client with the purpose of easy retrieval of data (Sjöö and Ungerbäck 2007). The client allows the users to perform searches for reports with limitations based on attributes, e.g. location and time. The locations of the results can be displayed on a map. Each report can be downloaded individually in PDF format for printing. As an alternative to the client, the Swedish Transport Agency can also provide analysts with copies of parts of database number 2 (Sjöö and Ungerbäck 2007). Appendix B is an example of a digitized, submitted and downloaded report. Figure 1.2 below is a screenshot of a section of a downloaded report, specifically the part that contains the illustration of the accident. Note that the illustration of the accident contains different symbology than what is used during the process of drawing by hand (see Appendix C). The symbology of the downloaded version of the report (Figure 1.2) is easier to comprehend by a larger variety of readers since the digital reproduction of the scene allows more detail.

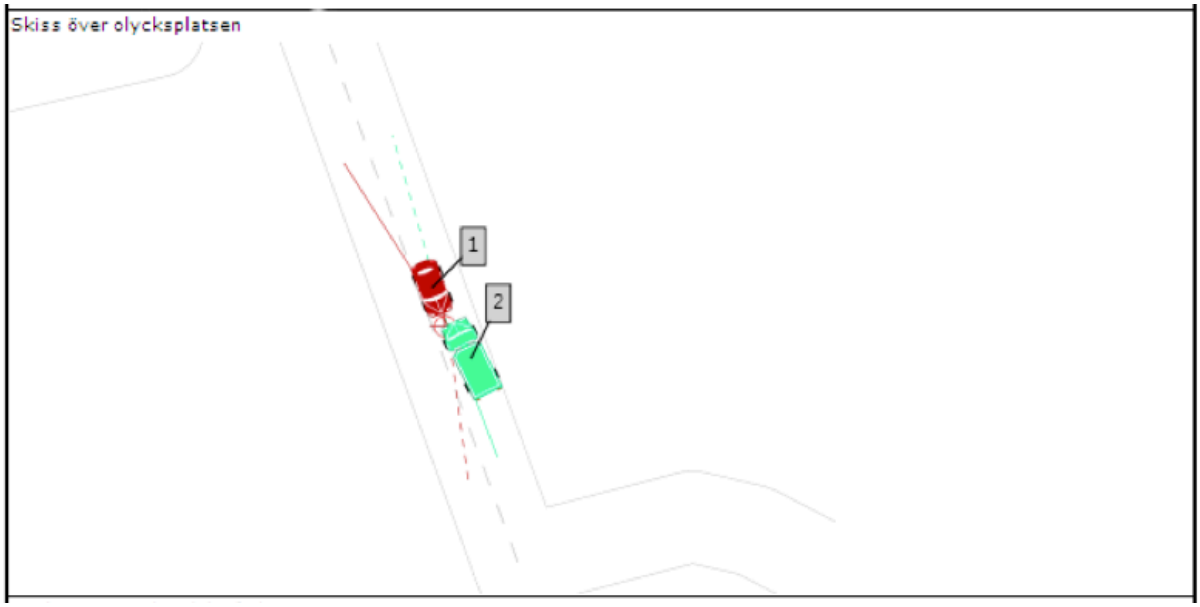


Figure 1.2. An illustration of the scene of an accident. Traffic elements are numbered so that they can be referred to in the text describing the scenario. Solid and dashed lines represent origin- and direction of travel. The crosshair marks the collision point (Mattsson and Ungerbäck 2013). A car, element number 1, has drifted into the oncoming lane causing a frontal collision with a truck, element number 2. Screenshot from a downloaded report in PDF format (Appendix B).

1.2.3 Problems with the current system

The Swedish Transport Agency has expressed concerns regarding the quality of data in STRADA. Sjöö & Ungerbäck (2007) mention three main shortcomings derived from the process of reporting: delayed submissions of reports (see Figure 1.3), reports with insufficient amount of information i.e. attribute data, and location inaccuracy.

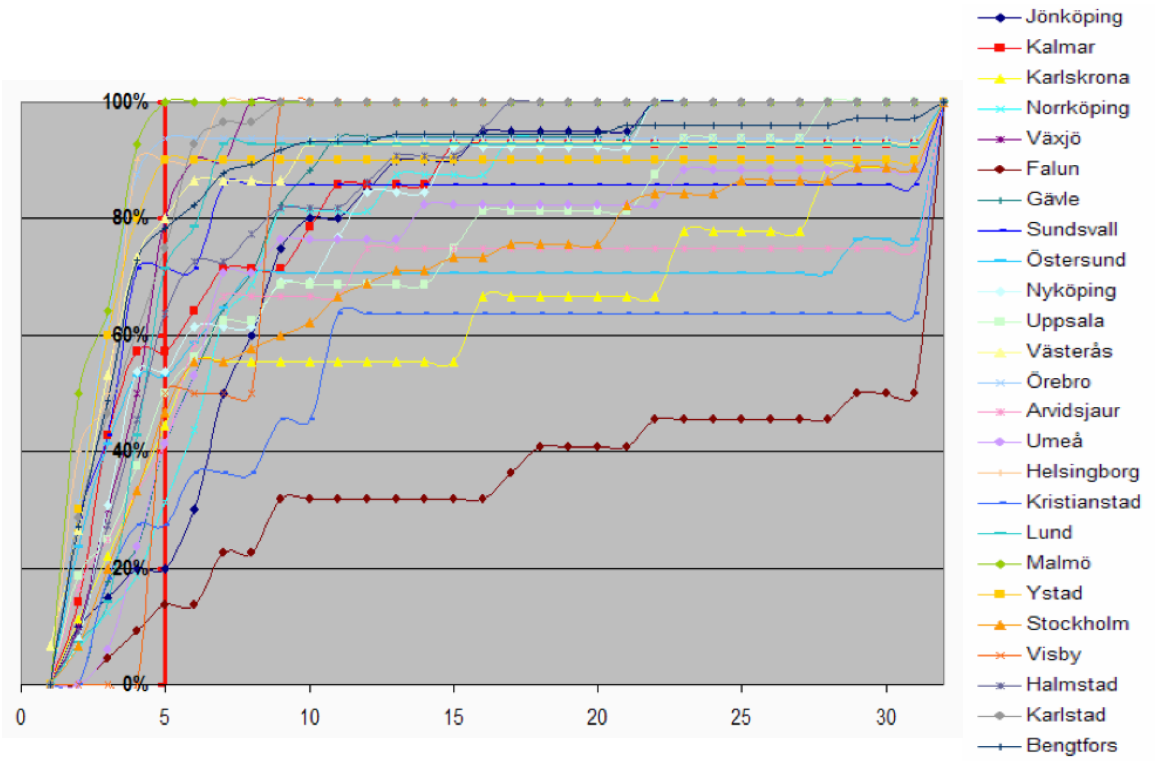


Figure 1.3. The proportion of reported fatal accidents in the year of 2005 by number of days from when police authorities were informed of them. The different colours represent different municipalities (Sjöö and Ungerbäck 2007). According to the graph only one municipality, Malmö, managed to uphold the regulations. It can be argued that this data is old but almost 10 years later, in the period of 2013-2015, the proportion of fatal accidents reported within the 5 days limit was 56% on a national scale (Andersson and Berg 2016).

The location of an accident is to be specified by city, where possible, and by the road on which it occurred together with the name and relationship to the nearest intersecting road, e.g. “Road 110, 300 metres south of the intersection with road 105”. Instructions mention that a GPS device should be used when possible to increase accuracy (Mattsson and Ungerbäck 2013). However it is unclear how often such a device is used. The paper form requires that the person completing it can draw and has handwriting that the person who will digitize it can read and interpret correctly. The data collection process has, since the beginning of STRADA been intended to be performed in situ with a portable computer and GPS device. This would, according to the Swedish Transport Agency, shorten the process of submitting reports as well as improving the quality of the data (Sjöö and

Ungerbäck 2007). Attempts to implement such a system have been made (Ungerbäck 2008, 24 April 2015) but cancelled (Sjöö and Ungerbäck 2007; Ungerbäck 24 April 2015). For several years Swedish police has been implementing an entirely new IT-system. It originates from a political decision to improve the efficiency of police work in order to let officers spend more time in the field and less time at police stations with administration which current systems demand. The intention was a more user-friendly system where the most common crimes could be reported quickly in situ with the help of a portable computer or laptop (Tallungs and Josefsson 2014) . The function of reporting traffic accidents to STRADA was intended to be incorporated into this larger system. However the implementation of the new system has been problematic, thus the new methods for police work, including digitized reporting of traffic accidents have been delayed or cancelled (Ungerbäck 24 April 2015; Tallungs and Josefsson 2014).

1.3 Aim

The aim of this project is to design and develop a mobile GIS for collection and management of accident information by police authorities.

The mobile application shall have functions such as:

- A map with the possibility to draw certain objects such as cars along with symbols required according to the guide (Mattsson and Ungerbäck 2013), e.g. direction of travel.
- Input elements for attribute data of the current form (see Appendix A).
- Operation of the camera of the device to capture photographs.
- The use of the GPS of the device to obtain coordinates.
- Internet connectivity and possibility to send data to a server.

1.4 Delimitations

The project does not solve problems of compatibility with larger systems such as the ones at Swedish police authorities.

Although the application and system should be easy to use, the project focuses mainly on the technical issues with creating it. No survey or analysis has been made regarding the optimization of the user interface, a task closer to the implementation of the

system. The resulting user interfaces are based upon the author's own opinions in relation to the literature.

The project does not include practical tests of the application with police officers.

2 Earlier research, examples and technology

The first step to determine the architecture of a suitable system is to study earlier examples. For facilitating disaster management, a topic arguably similar to the one of this project, Mansourian et al. (2006) propose a web-based system in an SDI (Spatial Data Infrastructure) framework for reliable information collection and sharing. The web-based system consists of the main components of a web GIS including user interface, web server and application server, map server, data server, and a database. A Web-based system would have a similar structure to what this project is looking to achieve excluding the extra user interface, which is the added mobile platform for data collection. Kresse and Danko (2012), present a similar architecture for a Web-GIS while describing the implementation of a map server. A map server generates maps from spatially referenced data and is an important part of a Web-GIS (Kresse and Danko 2012). The architecture described by Danko & Kesse is presented as a three-tier architecture containing a client, a server, and a geospatial database access system. The server tier consists of three parts: a web server, an application server, and a map server (Kresse and Danko 2012). The architectures described by Mansourian et al and Danko & Kesse are essentially very similar despite some differences in the way that they are presented. Most importantly, the descriptions of how the components interact and function together are the same.

The user interface, or client, is where data are presented for visual analysis. In a web-based GIS the user interface consists of a graphical environment viewed in a web browser. The graphical environment consists of several documents with HTML tags and JavaScript elements (Mansourian et al. 2006; Kresse and Danko 2012). The web server receives requests from the client in the form of a URL. The request is forwarded to a map server with the help of an application server. The map server will, as mentioned earlier, generate maps from spatial data based on the request coming from the web server. The spatial data processed by the map server is served by the data server which provides it from the database following a request (Mansourian et al. 2006).

Regarding the database, Mansourian et al (2006) recommend including mirror databases for backup of data to ensure that the system is always operational. The mirror databases are activated when problems occur with the original database. When not active, the mirror databases replicate all changes made to the original database. The user should not notice any difference depending on which database is currently active (Mansourian et al. 2006).

As a developer, part of the task of planning the architecture of a system is to select platforms and tools to work with. Having limited personal experience of e.g. application-making for mobile devices and of creating servers and databases, it becomes a requirement that resources such as extensive instructions and documentation are available for the chosen software and platforms. Additionally, the budget of the project is limited. The conclusion is that the architecture should consist of free and open source software (FOSS) and platforms to a greatest possible extent. The purpose of open source is to, with the help of an engaged community, collectively create and improve source code for a rapid evolution of software (Open Source Initiative 2012, n.d.-e). Source code is a programme in its original programming language, e.g. C or Java. It is the language used by persons writing the programme and will later be transformed for the machine to execute.

The open source label was established in 1998 after the public release of the source code of a web browser called Netscape (Open Source Initiative 2012). The organization Open Source Initiative (OSI) was formed in the same year for educational and advocacy purposes with a mission to explain and protect the open source label. Part of it was to set a definition for what should be considered to be open source (Open Source Initiative 2012). Today, the definition is a list consisting of 10 paragraphs. A large part of the open source definition deals with distribution of source code. The source code should always be distributed together with an open source programme. In case a programme is not distributed with source code there should always be a well-publicized method to obtaining it, preferably by downloading it online free of charge (Open Source Initiative n.d.-e). To ensure that evolution is maintained, source code is always distributed under a license which allows software to be freely used, modified, and shared (Open Source Initiative n.d.-c). The Open Source Initiative has a list of approved licenses that are recognized and generally referred to in the community (Open Source Initiative 2012, n.d.-c). The open source community has developed several open source tools relevant to this thesis. Some examples are MapServer and GeoServer, two software servers specialized in sharing geographical data (i.e. map servers), another one is PostgreSQL, a database system.

Since the establishment of open source, the function of the internet has developed as well with new concepts emerging, most notably the concept of Web 2.0 which was established at the Web 2.0 conference in 2004 (O'Reilly 2005). An essential part of Web 2.0 services or applications is database management. Unlike earlier ways of building

large databases, i.e. paying people to do it or getting volunteers to do it, the Web 2.0 way is to make the users, or customers also become contributors (O'Reilly 2005). Success on the Web 2.0 market is thus dependent on the network effects from user contributions and the value of software is proportional to the size and dynamism of the database it manages (O'Reilly 2005). Some examples representative of Web 2.0 are peer-to-peer services, web services, Wikipedia, Flickr, Google, YouTube, Facebook and folksonomy: the collaborative organization of information by users with the use of key-words, or tags, to build associations (O'Reilly 2005). Regarding maps, Web 2.0 map applications, e.g. Google Maps, Bing Maps and Esri's ArcGIS online, are more interactive than previously making them sometimes as flexible as desktop applications (Kresse and Danko 2012).

The interactivity and increased usefulness of map applications have, in the true spirit of the Web 2.0, enabled users to contribute to maps with their own data. Volunteered Geographic Information (VGI), the collecting of geospatial information by volunteers; professionals, or amateurs is a phenomenon derived directly from the evolution of the web (Goodchild 2007; Kresse and Danko 2012). OpenStreetMap (OSM) is perhaps the most well known example of VGI. In comparison to Google Maps, which is commercial but still contains VGI in the form of e.g. geotagged photos, OSM consists solely of data provided by volunteers.

Progress in mobile phone technology has made it easier for volunteers to become contributors of VGI. Everyone owning a smartphone is now also the owner of a GPS receiver. Location-based services (LBS); the intersection of GIS, internet and mobile devices (Kresse and Danko 2012), enables users to become sensors and is a powerful medium for collecting VGI (Tao 2010). Digital collection of data in the field is not a new occurrence. In fact, it has been used by professionals since the end of the 1980s (Clegg et al. 2006). An example is ESRI's ArcPad, a data collection software for handheld devices (ESRI n.d.; Clegg et al. 2006). LBS and mobile GIS offer quick submission and overview of data, which is why they are suitable for e.g. crisis management (El-Gamily et al. 2010). The devices for which the early digital data collection systems such as ArcPad had been created were PDAs (personal digital assistants) and tablet PCs. The recent introduction and popularization of smartphones has brought a large amount of applications for geographical data collection and mapping, e.g. ESRI's Collector for ArcGIS (see Figure 2.1) and FME Reporter, applications such as ESRI's Survey 123 for ArcGIS and ODK Collect, the latter open source, that offer customized digital geotagged forms, and Tyck Till (roughly "leave a comment" in Swedish) - a mobile application

released by the municipality of Stockholm for citizens to make error reports containing geographical information with possibilities for attaching media such as photos. Many applications follow the Web 2.0 concept and offer both viewing of data and data input, e.g. applications for recreational activities such as crowd sourced offline tourist maps, e.g. TripAdvisor, Geocaching, and applications for navigation, e.g. Google Maps and Waze where users submit data both passively and actively. Applications exist for offline use of OSM data on mobile devices, e.g. OSMand, and there are a number of applications, e.g. GPS Test and GPS Essentials, that explore the GPS of the smartphone showing coordinates, coordinate system, number of satellites, accuracy and much more so that the device can be used as a GPS receiver for field work.

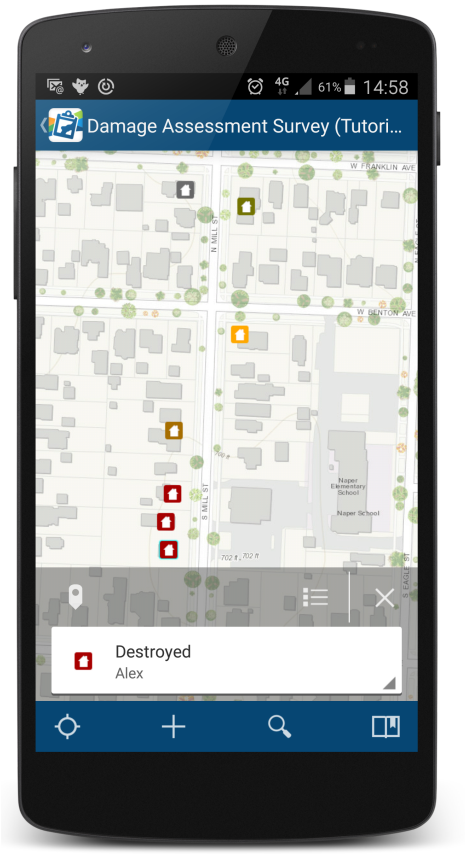


Figure 2.1. ESRI's Collector for ArcGIS.

The occurrence of many applications could be partially explained by the fact that the platforms on which they are built are either open source or made accessible in a similar way. An incentive to give away source code, apart from maintaining the evolution of a programme, is that others can create extensions and new tools based on the code of the initial programme leading to a larger market share for it as a product. The decision to go Open source can thus be a useful strategy when creating a platform (DiBona et al. 1999). The incentive is similar to the described concept of Web 2.0 where, to obtain user contributions, one must create an initial product used by many. In some cases a solution for increasing the popularity of a product is to distribute an application programming interface (API). An API is a collection of guidelines and specifications for a system that make programming easier for that particular system. Using API, programmers can write a programme without deep knowledge about the system itself (Merriam-Webster n.d.). Google did this early when they distributed an API for Google Earth making it useful to millions of users, a deed by some referred to as the democratization of GIS (Goodchild 2007). Now one can also use the API of Google Maps to embed and customize their maps on websites or mobile applications adding to the popularity of the application.

With large user communities, the evolution of the web and lower thresholds for creating customized tools it is now possible to build a system consisting, as desired, solely of free and open source software. An example is the proposed web-based prototype architecture for reporting pavement damages by Brovelli et al (2014). Client side, the data collection process is conducted on a mobile device while the output can be visualized on desktop computers, tablets, and mobile phones. This project however only requires data collection on mobile devices and output to desktop computers. The prototype architecture of Brovelli et al (2014) is similar to what this project is aiming to achieve. Since the author has prior experience of some of the components, the choice of components for the project has been based on this example. The open source components used by Brovelli et al are Android, ODK Collect, ODK Aggregate, Apache Tomcat, PostgreSQL, PostGIS, GeoServer, OpenLayers, jQuery mobile, and Leaflet. The components that have been chosen for the system created for this thesis are listed in chapter *3 Methodology and tools*. ODK Collect and ODK Aggregate, the set of tools that comprise nearly the entire data collection infrastructure of the prototype architecture proposed by Brovelli et al., are not flexible enough for an architecture that can fit the aim of this project. The alternative, which was done, is to learn how to create a mobile application from the very beginning.

3 Methodology and tools

The project started with a study of the current system of accident reporting to understand the requirements that a digital solution would need to fulfil. It was followed by literature reviews to gain understanding about the system of a possible solution. In parallel, a mobile platform for which the application was going to be made was chosen. The platform is Android (see 3.1. Android). The process then consisted of learning about the Android mobile platform, Java (the programming language used by Android) and application making. The beginning of the learning process consisted of literature studies together with exercises to gain understanding of Java, exercises from the book "Learning Android" (Gargenta 2011) to build a practice application, and examples from the training section of the Android Developer website (Android Developers n.d.-b).

Work on the actual mobile data collection application started with a requirement specification and planning of different layouts of the user interface. When the mobile application had been developed to a certain level work also began on a database. Ensuring that required data could be collected (the application) and stored (the database) was first priority during the realisation of the system. After successfully submitting data to a table in the database work followed with a map server and an HTML document where the output could be viewed during testing. The HTML document would later become the web-based client, or web service. The strategy was to initially keep every step simple and focus on establishing a successful flow of data through the system. Subsequently the system would be gradually improved with e.g. storage of new data types, more tables, complete user interfaces and increased functionality. A web server, the last component added, was added during improvement for the web service to be able to perform functions that were becoming increasingly complicated. The web server also enabled work on some security issues (see 4.2 Web server), however the report only deals with security on a basic level.

The chosen tools have more or less extensive manuals either bundled with the software or available online on official websites. Online communities of users that discuss problems and solutions, most notably Stack Exchange and Stack Overflow have been essential resources for the project as well. Below follows a list and information about the chosen software and tools.

3.1 Android

Android is an open source platform for mobile devices (Gargenta 2011). It is built on the open source operating system Linux (Gargenta 2011). Operating systems manage communication between software and hardware, e.g. Windows and Mac OS (The Linux Foundation n.d.). Android Inc. was acquired by Google in 2005. Many were expecting the announcement of a Google Phone but instead Google and a group of technology and mobile companies including, e.g. Motorola, Texas Instruments and Samsung, formed the Open Handset Alliance which would in turn release Android as an open source platform (Gargenta 2011; Open Handset Alliance n.d.). The purpose of the Open Handset Alliance is to accelerate mobile technology and user experience (Open Handset Alliance 2007). Unlike other mobile phone platforms, Android is not linked to a specific device. It was mainly designed to be portable and the clear separation between software and hardware enables it to run on a large variety of manufactured devices (Gargenta 2011). The strategy is to primarily create a platform used by many. Google, as the media company it is, then provides services on top of that (Gargenta 2011). Android is licensed under the Apache License version 2.0, which permits the user to e.g. modify, redistribute and sublicense it for public or commercial use (Android Open Source Project n.d.; The Apache Software Foundation 2004).

3.1.1 Android Studio

Android Studio is the official integrated development environment (IDE) for Android (Android Developers Blog 2014; Android Developers n.d.-a). Success in the competition for users between the largest smartphone platforms is dependent of the amount of mobile applications available for each respective platform. In order to gain market shares they all now offer their respective software, e.g. Android's Android Studio, Apple's Xcode and Windows' App Studio, designed to help ambitious users contribute by creating their own applications. Android can be considered as most successful in this respect as it is the platform with largest amount of available applications (Statista n.d.). Android Studio offers many useful tools for the development process including an emulator for testing on virtual devices.

The planned mobile application for data collection has been created in Android Studio version 1.1.0 (see 4.1 Android Application). As mentioned in *3 Methodology and*

Tools, the application is written in Java. Layouts, or user interfaces are, when not created programmatically, written in XML.

3.2 PostgreSQL & PostGIS

PostgreSQL is an open source object-relational database system compliant with all major operating systems including Windows and Mac OS (The PostgreSQL Global Development Group n.d.-b). It is provided under the PostgreSQL License, which can be described as a very liberal license. It grants permission to use, copy, modify, and distribute the software for any purpose, i.e. even a company could modify it for commercial use (The PostgreSQL Global Development Group n.d.-b; Open Source Initiative n.d.-f). In addition to supporting most SQL:2008 data types, e.g. integer, char, date, numeric, PostgreSQL can also store binary large objects (BLOBs) such as pictures, sounds and video in binary format (The PostgreSQL Global Development Group n.d.-b). Table 3.1 describes the limitations related to storage space. As shown, PostgreSQL is more than sufficient for the intended database. The only constraint that could become a problem in the future appears to be the one of maximum table size (32 terabyte) unless new tables are created when the current table is approaching maximum size.

Table 3.1. Storage limitations of the PostgreSQL database. Source: PostgreSQL (n.d.-b).

Limitation	Value
Maximum database size	Unlimited
Maximum table size	32TB
Maximum row size	1,6TB
Maximum field size	1GB
Maximum rows per table	Unlimited
Maximum columns per table	250-1600 depending on column types
Maximum indexes per table	Unlimited

PostgreSQL can interpret several languages such as Java, Python, C++ and PHP (The PostgreSQL Global Development Group n.d.-b).

PostGIS is an extension to PostgreSQL that enables support for geographical objects in the database (Refractions Research n.d.). This means that the database can store georeferenced points, lines and polygons for e.g. a web map service. PostGIS is offered under the GNU General Public License (GPL-2.0), which permits free copying, distribution and modification of the software code as long as it stays under the same license (Refractions Research n.d.; Open Source Initiative n.d.-b).

3.2.1 pgAdmin

pgAdmin is an open source administration and development platform for PostgreSQL. It may be used with several operating systems including the most common Mac OS X, Windows and Linux (pgAdmin n.d.). The software with its user interface provides an easy way to manage a PostgreSQL database and, e.g. create tables. With tools like the SQL Editor in pgAdmin the user can make SQL queries to the database. Like PostgreSQL, pgAdmin is also provided under the PostgreSQL License (see 3.2 PostgreSQL & PostGIS)(pgAdmin n.d.).

3.3 Apache Tomcat

Apache Tomcat is a free and open source application server and web server (Open Source Geospatial Foundation n.d.-d; The Apache Software Foundation n.d.). It is provided under the Apache License version 2.0 (see 3.1 Android) (The Apache Software Foundation n.d.). Apache, or Apache HTTP Server is the web server part of the software. It can also be installed separately on e.g. PostgreSQL database server. Apache can power websites and web services by hosting and sharing their web content in the form of HTML documents.

3.3.1 PHP

PHP is a recursive acronym for “PHP: Hypertext Preprocessor” (The PHP Group n.d.-b). It is a scripting language useful for its dynamics and can be embedded into HTML documents. Unlike JavaScript and HTML, which are processed client-side, PHP is processed server-side. The condition is that the web server on which a document containing PHP code is located has an installed PHP parser (The PHP Group n.d.-a). When a user is browsing a website and gets directed to a document containing PHP code,

the PHP part of the document will be processed by the parser on the server before sending the HTML part of the document to the client computer, i.e. the user. There is an exception for when one does want PHP to be visible to the user: the “echo” method. The method is a way of displaying output rather than actual operations. Figure 3.1 has an example of PHP code.

```
<?php
      $a = 10;
      $b = 6;
      $c = ' Hello';
      $d = ' World!';

      echo '20' . $a+$b . $c . $d;
?>
```

Figure 3.1. Example of PHP and echo method. The script is contained inside brackets that mark the start and end of PHP. In this example 4 variables are created with values assigned to them. The only part visible to the client is written in the echo method, which outputs the variables. The output of this script is ”2016 Hello World!”.

PHP is useful to make websites more dynamic and to e.g. handle the collecting of form data (The PHP Group n.d.-a). With PHP code, the developer can choose what to do with the data, e.g. send it to a database. Since the code is never visible to the client, it also hides possibly sensitive information such as connections to databases.

3.4 GeoServer

GeoServer is an open source software server for sharing spatial data, i.e. a map server (Open Source Geospatial Foundation n.d.-g, a). It comes with the Jetty web server on which it runs, or packaged as a standalone servlet (Open Source Geospatial Foundation n.d.-d), i.e. it extends the functionality of a web server or application server of choice, e.g. Tomcat (Oracle n.d.).

GeoServer allows sharing and publishing of spatial data in established open standards such as e.g. Web Map Service (WMS), Web Feature Service (WFS), and Web Cover Service (WCS). These standards are all OGC standards. OGC, or Open Geospatial Consortium, is an international non-profit organisation focusing on developing open standards for the geospatial community (Open Geospatial Consortium n.d.-a). It has hundreds of members in both private and public sectors (Open Geospatial Consortium n.d.-b). The function of the organisation is to reach consensus about the standards in the community (Open Geospatial Consortium n.d.-a). It also acts as a central organ and

reference. Consensus in standards is necessary to make geographical data interoperable which in turn is necessary for distribution. GeoServer can read data from many different spatial data sources, including PostgreSQL/PostGIS, and output to many formats (Open Source Geospatial Foundation n.d.-c). Just like PostGIS GeoServer is also free and offered under the GNU General Public License (see 3.2 PostgreSQL & PostGIS).

3.5 OpenLayers

OpenLayers is an open source JavaScript based library, or API for creating interactive map elements in the HTML environment. OpenLayers can display and combine geographical content from different sources, e.g. a WMS generated by GeoServer as a vector layer on top of a base layer from OSM or Microsoft's Bing Maps. OpenLayers is licensed under the BSD 2-Clause License (OpenLayers 3 n.d.), which permits modification, redistribution and commercial use as long the license is provided including the original copyright notice (Open Source Initiative n.d.-a).

3.6 Google Maps Android API

Google Maps Android API enables integration of Google Maps into a mobile application. It is a library similar to OpenLayers with the main exception that it also provides access to Google's map data, or base layers with worldwide coverage.

The reason for choosing to implement Google Maps API is that it, as mentioned, features base layers with worldwide coverage. The implementation of Google Maps API into an otherwise open source application can cause issues. The use of Google Maps API for Android is regulated through Google's terms of service. To be able to incorporate Google Maps into an Android application, a key is obtained from Google together with a license. The use of the API in an application is free as long as it does not exceed a certain amount of requests per time period. Google and the developer can both monitor the amount of requests. Two other requirements for using the Google Maps API for Android are 1. The application should be free of charge for the user and 2. It should be available to the public (Google Developers 2015). The last requirement especially cannot be fulfilled by an application such as the one in this report. There are exceptions: as long as the application is available to the public in e.g. the Google Play Store free of charge, one can limit users by requiring them to log in to the application to use it (Google Developers

2015). However, for security reasons it could be problematic to make an application used by police and other government organizations available to the public. It is possible to make exceptions to the license by entering a separate written agreement with Google or by obtaining their written permission.

3.7 Additional tools

3.7.1 Eclipse

Eclipse is an open source IDE (The Eclipse Foundation n.d.-a). Within the project, it is used for running the Tomcat application server.

3.7.2 Mozilla Firefox & Mozilla Firebug

Mozilla Firefox is an open source web browser used for testing during the development of the web service.

Mozilla Firebug is an extension to the Mozilla Firefox web browser. Firebug enables live monitoring of e.g. HTML and JavaScript (Mozilla n.d.). It has been used for debugging and monitoring during development and testing.

3.7.3 Notepad++ & Sublime Text 2

Notepad++ is a free text and open source code editor released under the General Public License (GPL) (Ho n.d.). It supports many languages including, e.g. HTML, XML and JavaScript, which the project requires editing of. Notepad++ runs on the Windows operating system (Ho n.d.).

Sublime text is a text editor for code and markup language (Sublime Text n.d.) similar to Notepad++ but for Mac OS X.

4 Design and implementation of the system

Figure 4.1 illustrates the general architecture of the system. As the figure shows, the system consists of an Android application for data collection, a database server with a database for storage, an application server/web server to host a software server (map server), a web server with PHP that handles requests and hosts a web service with a map for viewing and retrieving data.

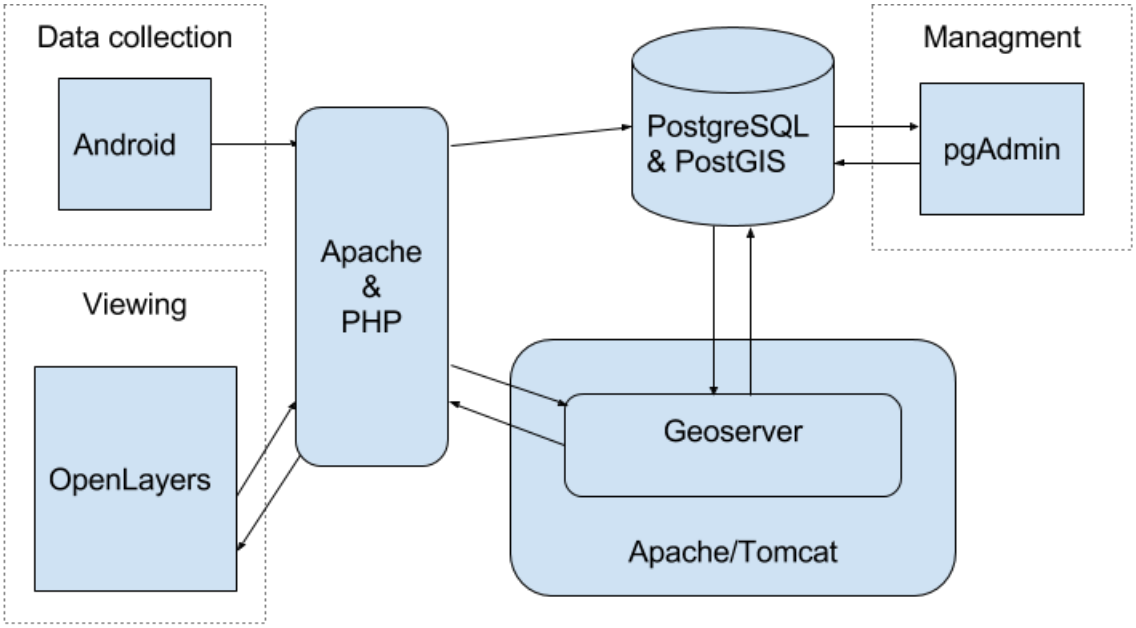


Figure 4.1. Architecture of the created system.

4.1 Android application

The application has been created for the Android 5.1 platform, or API version 22, on the Google Nexus 9 tablet, i.e. it works on other devices with API 16 or higher but the layout of the graphical user interface can look different on units with different screen dimensions. There is currently no need for alternative resources such as other languages or layouts that adapt to different screen sizes, resolutions and dimensions, since the application is designed for a specific homogenous group of users provided with the same type of device.

The application consists of a main activity and tabs for navigation between four fragments containing the map, the audio recorder, the camera, and the input for additional attribute data. Navigation between fragments is done by tapping their respective symbols

in the tabs of the top part of the screen or by swiping left or right. However swiping is used to pan while the map tool is in view.

The user starts the application at the scene of the accident. The first fragment to open is the map fragment (see Figure 4.2), equivalent to the drawing field or box 57 in the paper form (see page 1 of Appendix A, Figure 1.2 or Appendix B). The map tool is built on Google Maps Android API (see 3.6 Google Maps Android API). As seen in Figure 4.2, controls and tools are displayed on top of the map. These are zoom controls, a location button, save button, clear button, layer switcher, and the object switcher for drawing. The user will start by navigating the view of the map to their geographic position. The fastest way to do so is by tapping the location button in the upper right corner, which uses the GPS of the device to search for the location and, as seen in Figure 4.2, centers the view on those coordinates. For increased accuracy, it is also possible to navigate manually by holding down and dragging a finger on the screen. This is particularly useful in situations where the user will conduct the reporting process further away from the scene due to local conditions.

Figure 4.2. GPS of the device has located the position of the person holding it. It is represented as a blue point on the sidewalk at the intersection.

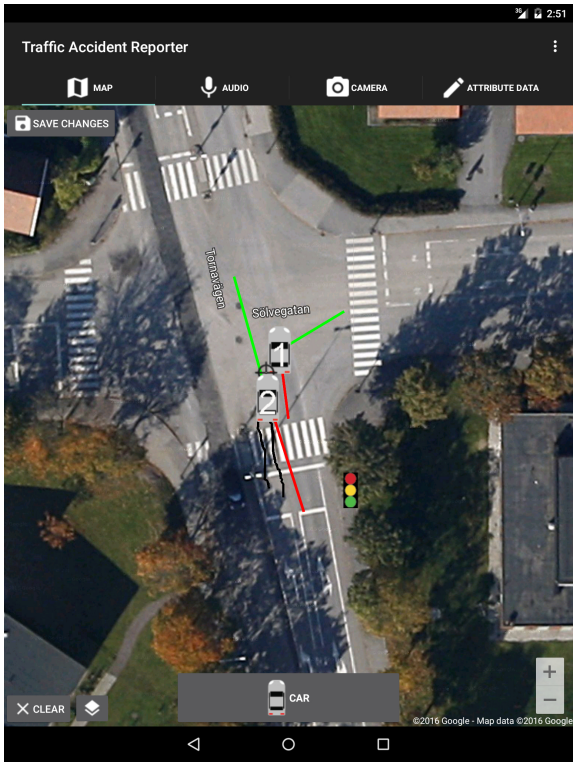


Figure 4.3. An example of a drawn scenario.

After finding the location, the scenario of the accident is described visually by drawing it. In the example of Figure 4.3, the first object added was car number 1. Since the object switcher was set to “car” it was added simply by tapping the screen once. There is also the possibility to move drawn objects. The drawn example in Figure 4.3 and 4.4 shows two cars; car number 1 and number 2, their respective origin of travel as a red polyline, direction of travel as a green polyline, skid marks from car number 2 as a black polyline, the point of collision as a crosshair, and a traffic light. The application aids the user with the order in which objects shall be drawn by selecting new symbols for the object switcher as well as providing short instructions. The user can also pick objects to draw manually from a library, which is displayed by tapping the object switcher button (see Figure 4.5). The traffic signs in the library are resized versions of Swedish traffic signs available online. These symbols are in the public domain and cannot be copyrighted, thus free to use. Other objects, or markers, as they are referred to by Google, are designed by me, the author of this report.

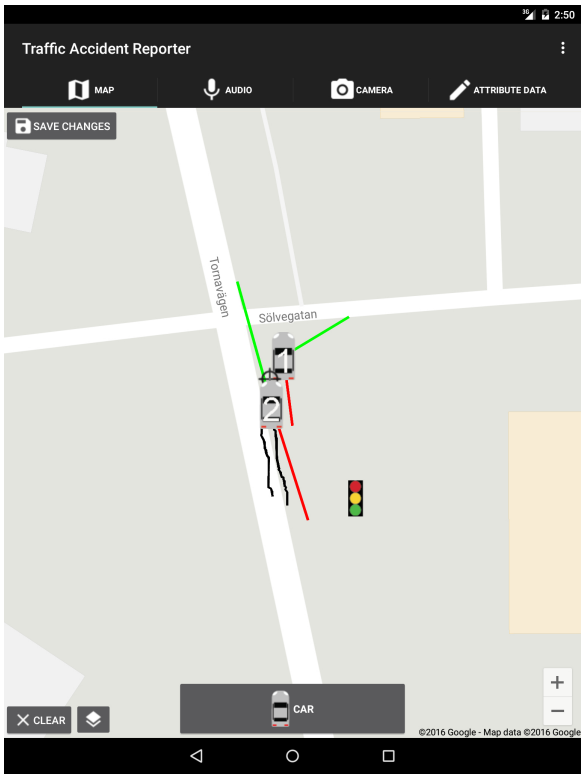


Figure 4.4. The base map has changed from hybrid to normal after tapping the layer switcher button.

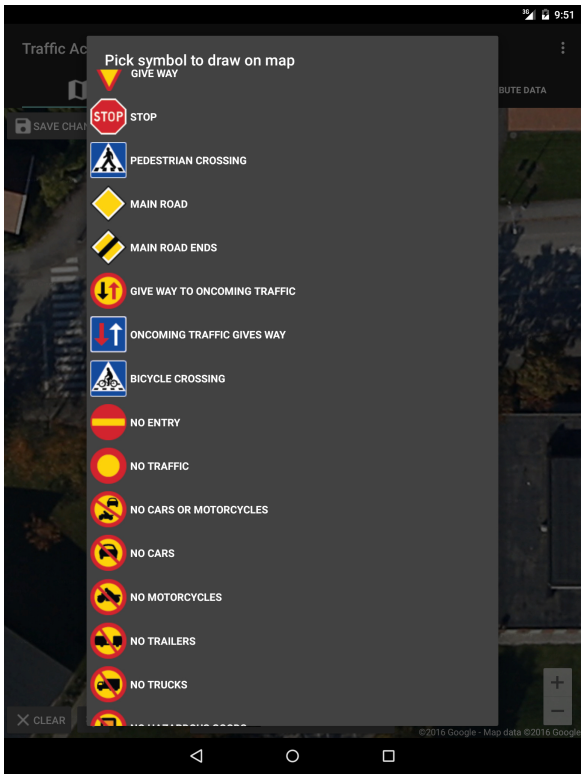


Figure 4.5. The library of objects is scrollable and contains many symbols.

When the drawing is finished and saved, the user continues to the next fragment, the audio recorder or dictaphone (see Figure 4.6). Here, information can be added in the format of an audio recording by utilizing the microphone of the device. The menu contains controls for recording, stopping, playing the recording, and deleting it if desired. After audio has been recorded, the user will continue to the next fragment, the camera fragment (see Figure 4.7). The first layout of the camera fragment (see Figure 4.7) contains a button that will start the camera of the device as seen in Figure 4.8.

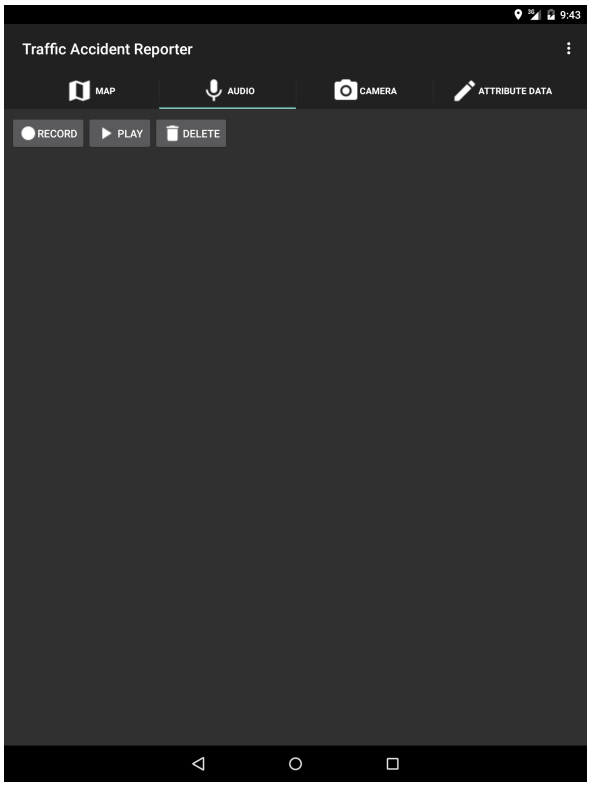


Figure 4.6. The audio recorder fragment.

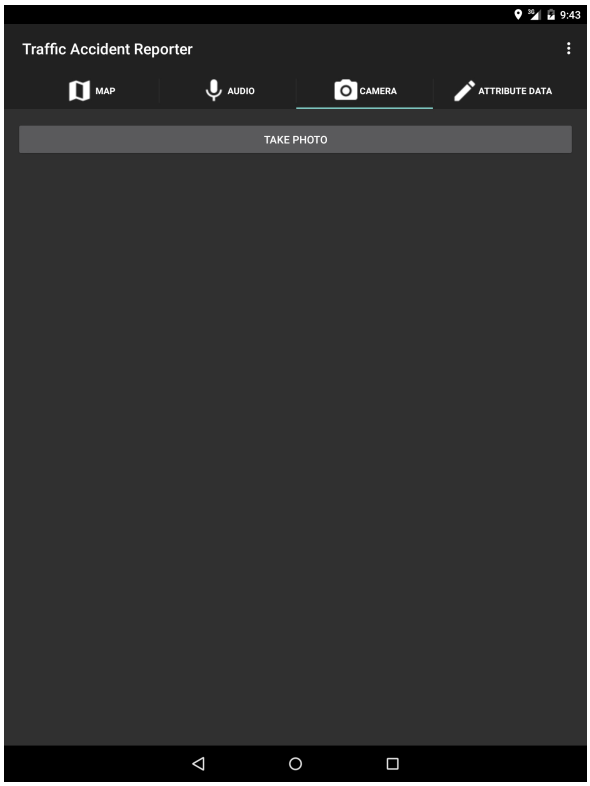


Figure 4.7. The camera fragment. A tap on the "Take photo" button will start the camera.

Figure 4.8. A photo has been taken. One can either retake the photo, cancel, or continue to the initial camera fragment with the new photo.

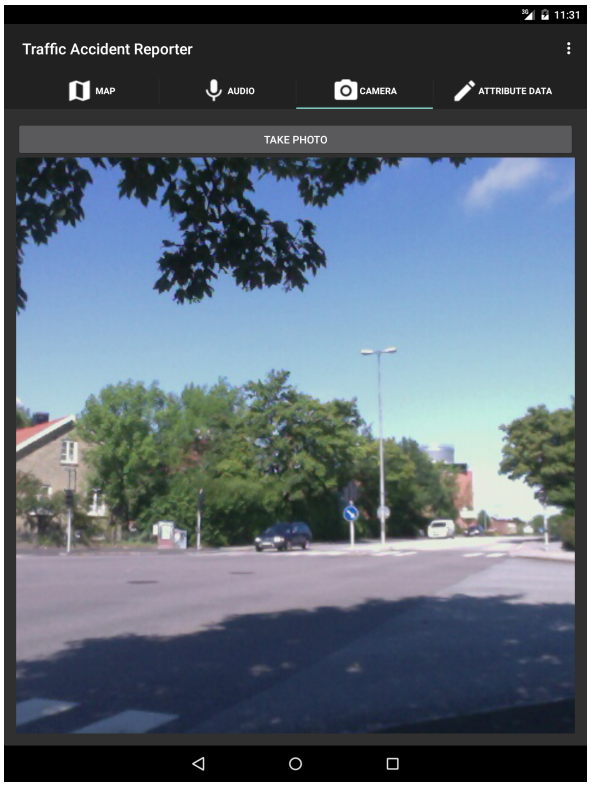


Figure 4.9. Back at the camera fragment with a photo taken by the camera.

After adding a photo (see Figure 4.9) it is time to continue to the last fragment titled "Attribute data" (see Figure 4.10). The layout of this fragment is a digital version of the rest of the paper form (see Appendix A). It contains EditText elements (fields for typing text), spinners (drop down lists) with several options, date and time picker (see Figure 4.11), and a submit button. Attributes are numbered for comparison with the text- or tick boxes in the paper form. Time and date are set in pop-up menus that appear after tapping the time- and date picker buttons (see Figure 4.11). The initial time shown in the pop-up menu will be the current time of the clock of the device. It can be accepted or changed. When scrolling down the attribute data fragment (see Figure 4.12), the user will discover that the application has created input fields for the two traffic elements that were added in the map, i.e. cars in this case. Some of their attributes such as element number and type have already been determined. Each one has been added to the layout programmatically while adding their respective objects to the map in the first fragment. Below, the user can input information about persons involved that were injured. By tapping the "add person" button input fields are generated for a new person. They can also be removed by tapping the "remove person" button.

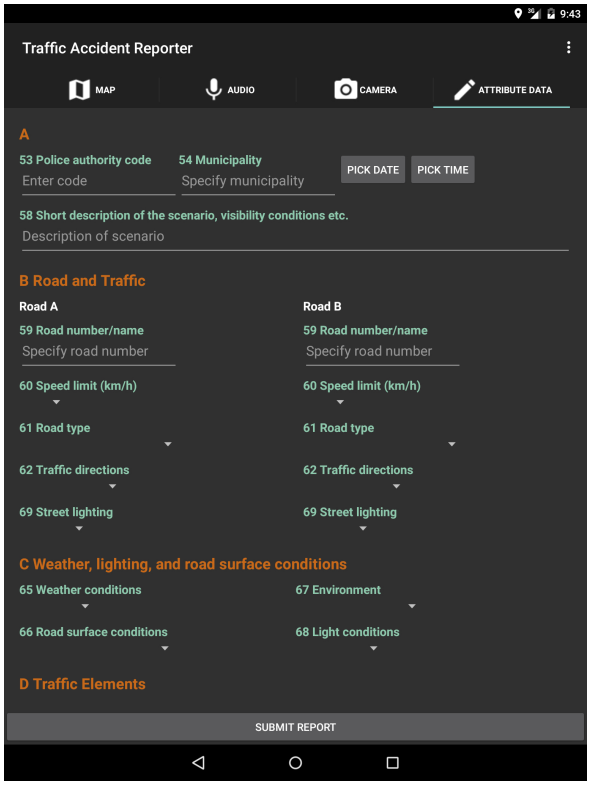


Figure 4.10. The attribute data fragment.

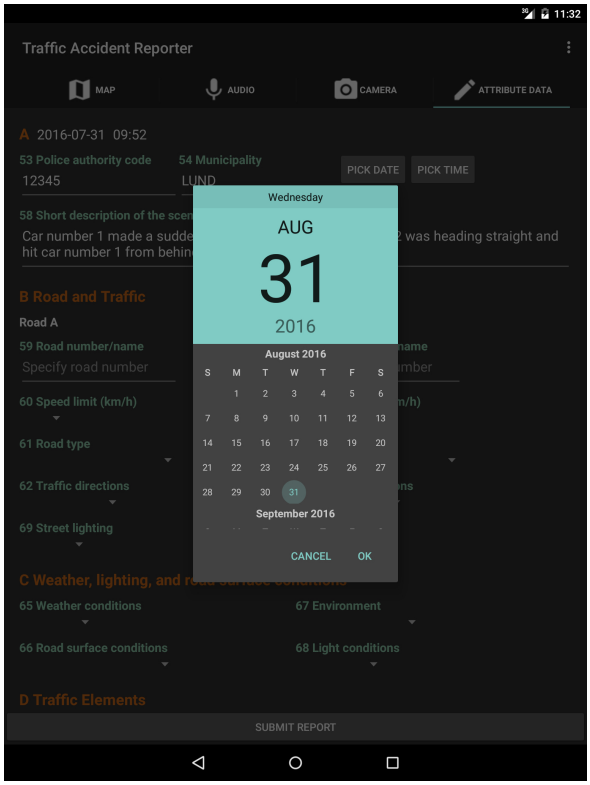


Figure 4.11. Date and time for the accident is set and added to the attribute data.

When all steps are completed, the user taps the submit-button at the bottom of the layout of the Attribute data fragment. The application will then connect to the Apache web server, encode images and audio to Base64 strings, get the values of all views, i.e. spinners, EditText elements etc., together with variables such as coordinates and, with POST methods, send them to a PHP document on the web server.

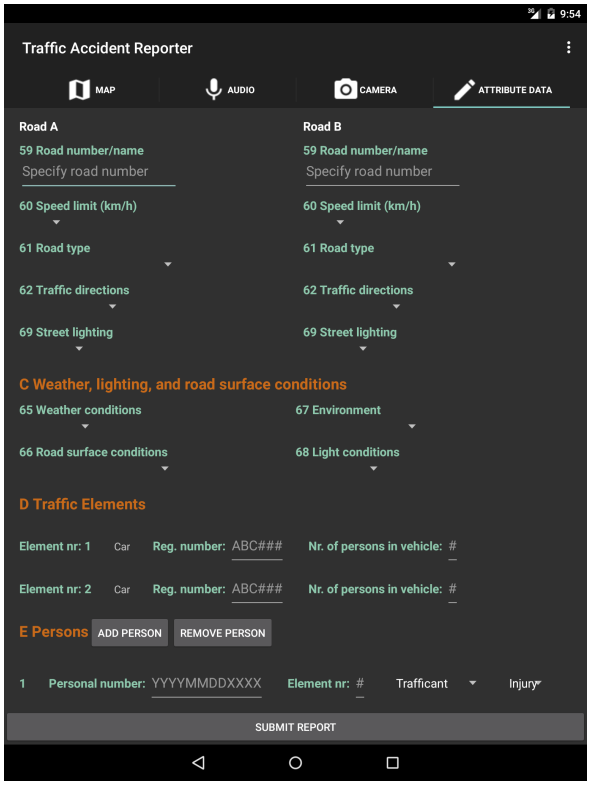


Figure 4.12. The bottom of the attribute data fragment. Input elements generated after adding the two cars to the map can be seen under the header titled Traffic Elements. Input elements for one injured person have also been generated after tapping the "add person" button.

4.2 Web server

As seen in Figure 4.1, 4.14 and 4.20, the web server is the mediator between the Android application, which the user is controlling, and the database. A user will never communicate directly with the database unless it is in the role of an administrator with pgAdmin. The implemented web server is a package with Apache and PHP (version 2.4.18-5.5.32-1) preconfigured for use with PostgreSQL. It has been downloaded and installed through Stack Builder, a tool for easy installation of extensions and tools related to PostgreSQL.

Several PHP documents have been created and are hosted on the web server. The PHP document to which the Android application sends the data as POST methods is one of them. It instantiates another PHP document which establishes a connection to the PostgreSQL server and database. With an established connection the script of the first PHP document will assign the value of each incoming POST method to a variable. Next, it formulates an SQL statement (see simplified example in Figure 4.13), which is sent to the connected PostgreSQL database server.

```
"START TRANSACTION;
INSERT INTO table_1_name (column)
VALUES ('$variablename');
INSERT INTO table_2_name (column)
VALUES ('$variablename');
COMMIT;"
```

Figure 4.13. An SQL statement containing a transaction method for inserting data into several tables (two in this example).

The statement contains a transaction method that inserts data into several tables. The reason for using transaction is that it ensures that either all sub queries in the transaction are carried out or, in the case of e.g. a syntax error, none are carried out. This prevents storage of corrupted data, e.g. inserting data into one table while inserting data into the next one fails. The variables containing attribute data about the accident are used to add values to the query. Since reports will contain varying amounts of traffic elements and persons, the query is dynamic. Loops extend the query depending on the amounts of elements and persons. Finally, the connection is closed. Figure 4.14 summarizes the explained steps performed by the web server.

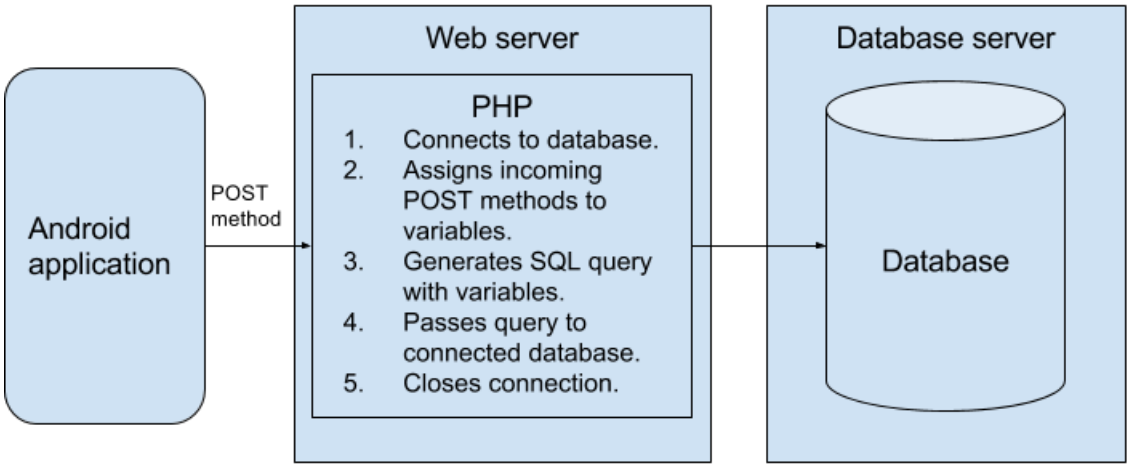


Figure 4.14. The function of the web server during the submission of a report.

Initially, the Android application communicated directly with the database server through the JDBC plugin which enables Java programs to communicate with PostgreSQL/PostGIS (The PostgreSQL Global Development Group n.d.-a). This is not recommended for security reasons and was subsequently changed. Communicating directly with the database server means that its address, user login and password as well as the SQL queries have to be stored in the code of the Android application. With PHP, this information is moved server side, away from the hands of the user. The database and the data stored in it are valuable and sensitive. It should be protected so that only complete and correct queries are made to it. Safety against incorrect queries and SQL injection attacks on the database can be added in the PHP document by adding constraints regarding values and characters that are allowed in a query. Functioning as a sort of filter, the web server takes some load off the database. Secondly, the work process while developing becomes less difficult since the queries are easier to manage in a PHP document than as one long string in java. Lastly, while testing during development the emulator in Android Studio does not require restart of the application each time since changes are not made to the code of the application.

4.3 Database server

The implemented database server is PostgreSQL 9.4 with PostGIS extension. The database stores all the data of submitted reports. It should be able to answer queries such as:

- Which accidents occurred in Lund municipality?
- How many accidents occurred at night?
- Who was involved in a specific accident?
- Which cars have been involved in accidents?
- Which are the accidents where a specific person has been the driver of a vehicle?
- How many fatal accidents have occurred in Lund municipality?

Although most of these examples are not used in the web service within the project, it is expected that these queries would be made in reality since they provide users of the data, e.g. police, with useful information. The database schema should thus be flexible enough to facilitate them. The resulting database schema (see Figure 4.15) is designed to be able to store all of the attributes while minimizing the amount of empty cells. It contains a database with 8 tables. The main table is named “accidents”. It contains the main attributes that a web service with a map would base queries on, e.g. municipality, weather, and light conditions. Some attributes such as scene and audio only occur once per accident report but are still stored in separate tables. The reason is that they both contain space consuming qualitative data that slows down the response time when a query is made to the database. Storing them separately makes the main table, “accidents”, quicker to process. The tables with more space consuming entries will only be processed if the user specifically wants to access them. It is also worth noting that the maximum table size constraint of 32 terabytes per table (see 3.2 PostgreSQL & PostGIS) becomes more manageable with this schema.

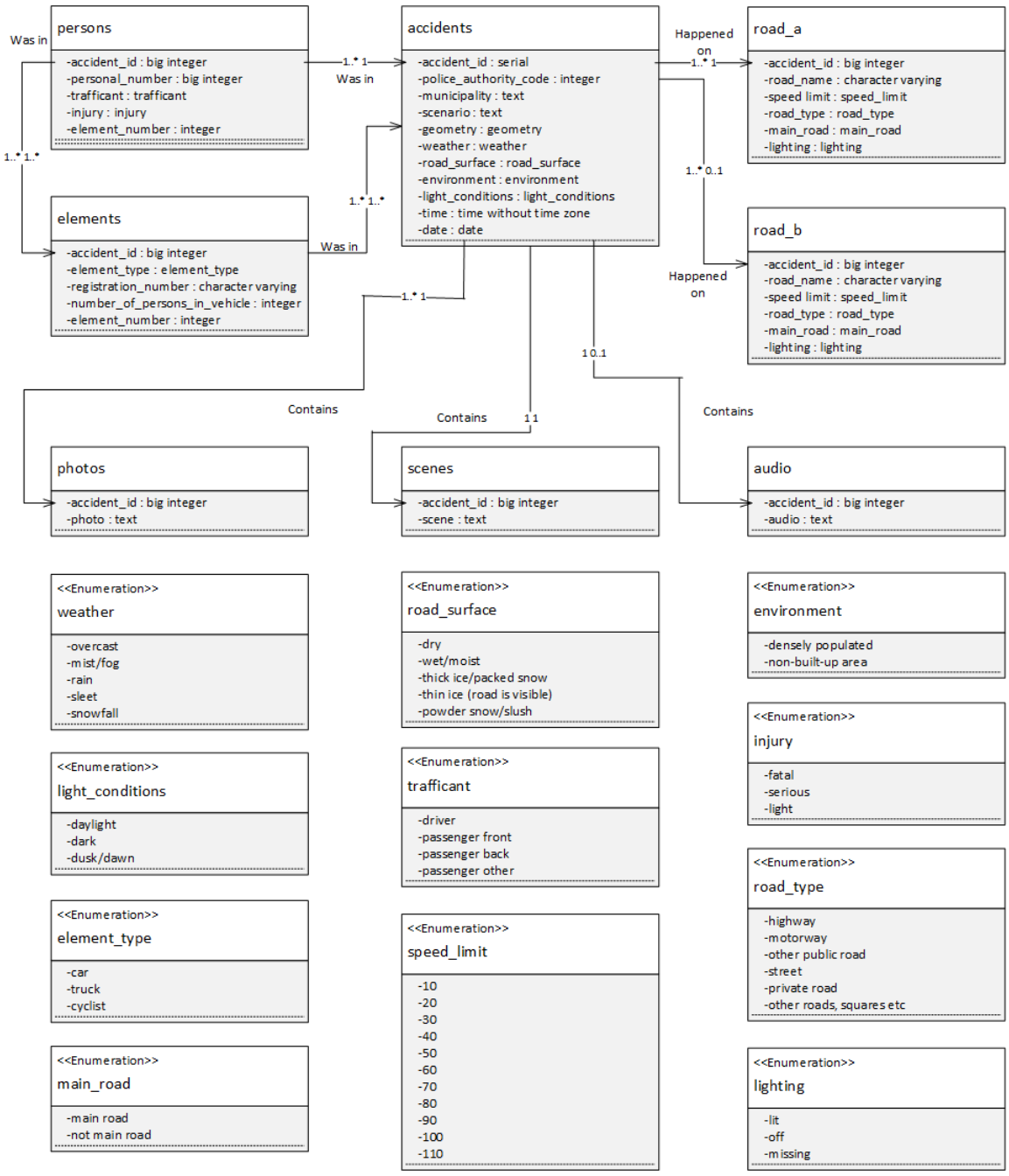


Figure 4.15. UML class diagram of the created database. Each enumeration represents a spinner in the Android application (see 4.1 Android application).

4.4 Application server and software server (map server)

The implemented application server is Tomcat 7, which is run with the help of the Eclipse IDE. In order to share the geographical data stored in the previously mentioned database (see 4.3 Database server), GeoServer version 2.7.1.1 is installed as a web application on the Tomcat application server. There are two reasons to why GeoServer has been installed as a web application on Tomcat instead of downloading and installing the alternative bundled with Jetty application server mentioned in *3.4 GeoServer*. Tomcat is the application server on which GeoServer has been tested the most and is thus the recommended alternative by the GeoServer manual (Open Source Geospatial Foundation n.d.-d). Secondly, and mainly, the settings of the application server had to be modified by adding a cross-origin resource sharing (CORS) filter in order for the web service (see 4.5 Web service) to function properly. Cross-origin resource sharing is when a request on one source of origin is made to get resources from another source. This includes different ports on the same computer. For security reasons web browsers by default apply some restrictions, or a same-origin policy for requests within scripts (Van Kasteren 2014; The Mozilla Foundation n.d.). The CORS filter allows the developer to make certain exceptions to those restrictions. The most extensive instructions for implementing such a filter were available for Tomcat.

A workspace has been created on GeoServer for the project. It contains a new vector data source, a "store", as GeoServer refers to it, connected to the PostgreSQL/PostGIS database. Finally, a new layer named "accidents" has been created using the new source. This is the layer used in the map of the web service (see 4.5 Web service).

The coordinates of each report are stored in the PostGIS database in the geodetic reference system WGS84. In order for the new layer to be displayed correctly on a map, the coordinates retrieved from the PostGIS database need to be transformed into a projected coordinate system. This is done in the layer settings in GeoServer. The output is WGS84 with a so-called "Pseudo-Mercator" projection (GeoServer version 2.7.1), generally referred to as Web Mercator. Web Mercator is the most dominant projection among web mapping applications and is used by e.g. Google Maps and Bing Maps (Jenny 2012; Finn and Usery 2014).

4.5 Web service

The web service created to view and download reports has functions similar to the tool which accesses STRADA database number 2 (see 1.2.2 Reporting to STRADA). Bootstrap 3, a library of CSS and JavaScript for web design has been implemented to improve aesthetics and functionality of the layout. Bootstrap is offered under the MIT license which permits anyone to "use, copy, modify, merge, publish, distribute, sublicense, and or sell copies […]" without warranty (Bootstrap n.d.; Open Source Initiative n.d.-d). jQuery version 1.11.2, another open source JavaScript library under MIT license (The jQuery Foundation n.d.), is required by Bootstrap and has been implemented as well. As seen in Figure 4.16, the user interface contains a map for overview of the geographical distribution of accidents. The map element has been created with OpenLayers 3 API. The base maps are from Bing Maps and OSM. An OpenLayers layer switcher enables changing of the base layer and the style of the vector layer. The point vector layer representing locations of accidents is the layer created in GeoServer published as a WFS. With OpenLayers 3 API the layer is retrieved from GeoServer through a URL containing a GetFeature request.

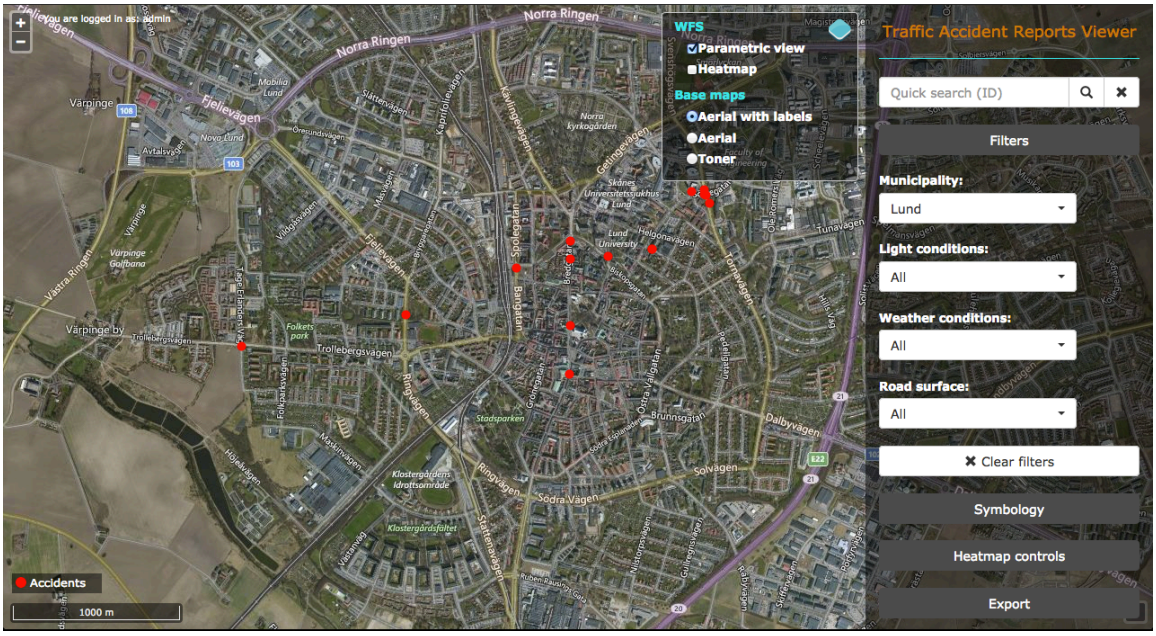


Figure 4.16. The web service. In this screenshot filters are set to display reports made in the municipality of Lund. The "Filters" sub section of menu is expanded. Filters can be combined for more specific search.

The initial plan was to publish the data in the WMS standard. In a WMS GeoServer processes data and generates a map which is sent to the client as an image (Open Source Geospatial Foundation n.d.-f; Open Geospatial Consortium n.d.-c). Advantages are e.g.

the possibility to style layers in GeoServer with SLDs (Styled Layer Descriptors) and to create legends. It is a good method for publishing a complete map. In a WFS, the geographical data, i.e. the features, are sent to the client in GML format to be rendered by OpenLayers client side (Open Source Geospatial Foundation n.d.-f), i.e. there is no legend or SLD from GeoServer. The decision to choose WFS standard for the project has been motivated by the fact that WMS does not allow selective retrieval of attribute data in a flexible way. A WMS delivers all the attribute data for one feature together using the GetFeatureInfo operation. The delivered data can be ordered in a list with freemarker template files and it is also possible to set the attributes that should be received by adding "propertyName:attributename" to the WMS URL. However, the values of the attributes in the returned data cannot be identified independently. The GeoServer manual mentions problems of flexibility as well and recommends the GetFeature method of WFS over the WMS GetFeatureInfo method whenever possible (Open Source Geospatial Foundation n.d.-e). The WFS standard is flexible and permits selection of attributes by their names to use them freely in the HTML environment.

The menu with the Bootstrap layout in the right end of the screen contains a number of controls: quick search, filters, symbology, heat map, and export. All of the buttons except for those belonging to the quick search field expand the menu and provide more controls when clicked. Filters enable displaying of accidents based on their attributes, e.g. limiting search, or GetFeature request to a certain municipality or a certain weather condition. Symbology controls enable the possibility to base the classification of symbols on attributes for e.g. visual comparison of distribution of different light conditions. The legend, which is written in SVG (Scalable Vector Graphics), will update its appearance accordingly. The layout of the web service is created mainly to demonstrate what a user interface could look like, test the flexibility of the collected data and explore its possibilities.

The top of the menu features an option to perform a quick search for single accidents based on ID value. Selective viewing of data from the database (as seen in Figure 4.16) is made possible by the addition of a CQL filter to the URL of the GetFeature request. A CQL filter consists of parameters written in CQL (Common Query Language) or ECQL (Extended Common Query Language) (Open Source Geospatial Foundation n.d.-b). The example URL in Figure 4.17 contains a CQL filter which limits the retrieved data from the database to reports with the value "LUND" in their "municipality" column (see Figure 4.15), i.e. accidents reported in the municipality of

Lund. In the web service, a URL similar to the example has been made dynamic by inserting variables that get their values from the combo boxes, or drop down lists, from the layout into the CQL filter.

```
"http://example.com/geoserver/reporter/ows?service=WFS&version=1.0.0&request=GetFeature&typeName=reporter:accidents&CQL_FILTER=municipality='LUND'&outputFormat=application%2Fjson"
```

Figure 4.17. Example of a URL containing a GetFeature request with a CQL filter (highlighted in gray) for the ”accidents” layer. The request will return features with the value ”LUND” in their ”municipality” column in the database table (see 4.3 Database server).

By clicking a red point, i.e. the symbol of a reported accident on the map, a modal element, or popup, appears displaying qualitative data, i.e. the attribute data of the accident (see Figure 4.18).

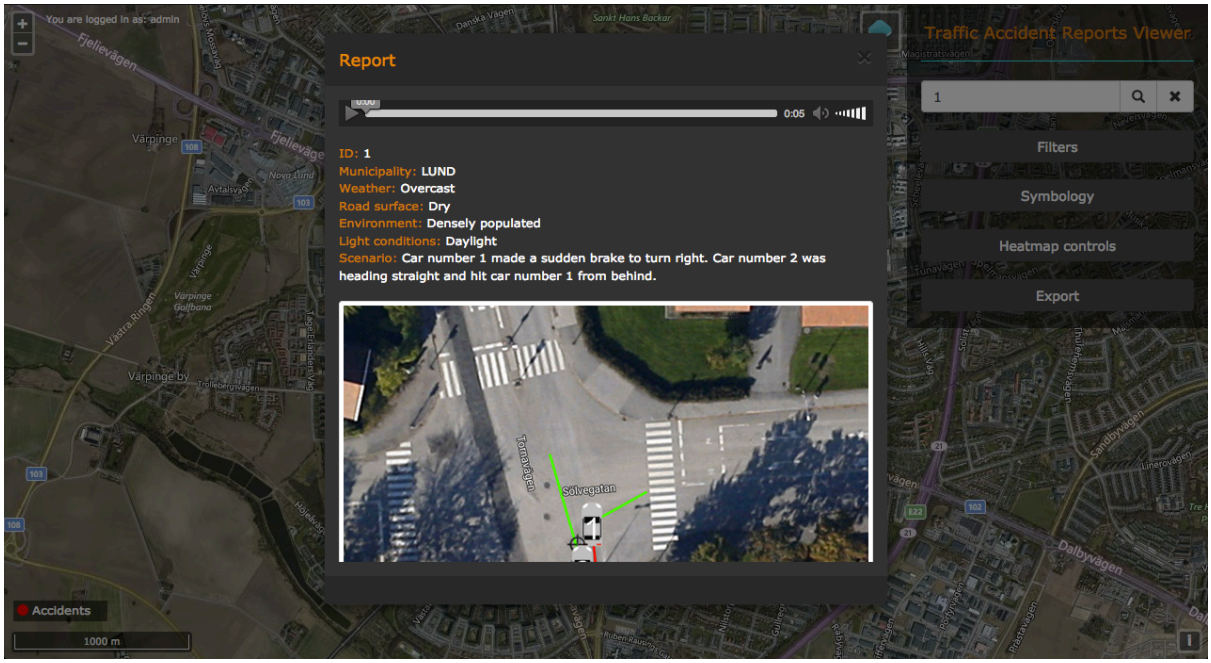


Figure 4.18. A report (red point on the map in Figure 4.16) has been clicked. A scrollable popup, or modal, appears with attribute data.

The attribute data displayed in the modal is retrieved by PHP. As stated earlier in the class diagram (see Figure 4.15), attribute data comes from several different tables in the PostGIS database. PHP script connects to the database and makes a request to select data associated with the ID of the clicked report. Since, as mentioned in 3.3.1 PHP, PHP is processed server side before the HTML is sent to the user, the body of the modal element consists of a separate PHP file on the web server. The separate PHP file is processed

again each time a modal element loads, which is when a feature on the map is clicked. Consequently the contents of the modal element become dynamic, i.e. dependent of which point is clicked. The output of the PHP is displayed in the modal with the earlier mentioned echo method (see 3.3.1 PHP).

During the process of creating the web service, a layer of the heat map class of the OpenLayers 3 API has been added while exploring the possibilities of data visualization (see Figure 4.19). The heat map has controls for adjustment of its symbology, namely blur and radius size.

Below the heat map controls, the menu contains an expandable sub menu with export-options (see Figure 4.19). The web service offers the possibility to export the view as a PNG image and the requested features as an Excel spreadsheet, or as a shapefile. An Excel plugin has been installed on GeoServer to facilitate the downloading of reports as Excel documents.

Figure 4.19. Reports displayed as a heat map on a grayscale base map for increased visibility. The base map is called Stamen and consists of OSM data styled by Stamen Design. The sub menus for heat map controls and export options are both expanded.

As a summary of the results, a UML sequence providing an overview of the system and describing interactions between components is presented in Figure 4.20. The figure contains a sequence showing the process of data input as well as a sequence showing data retrieval for the web service.

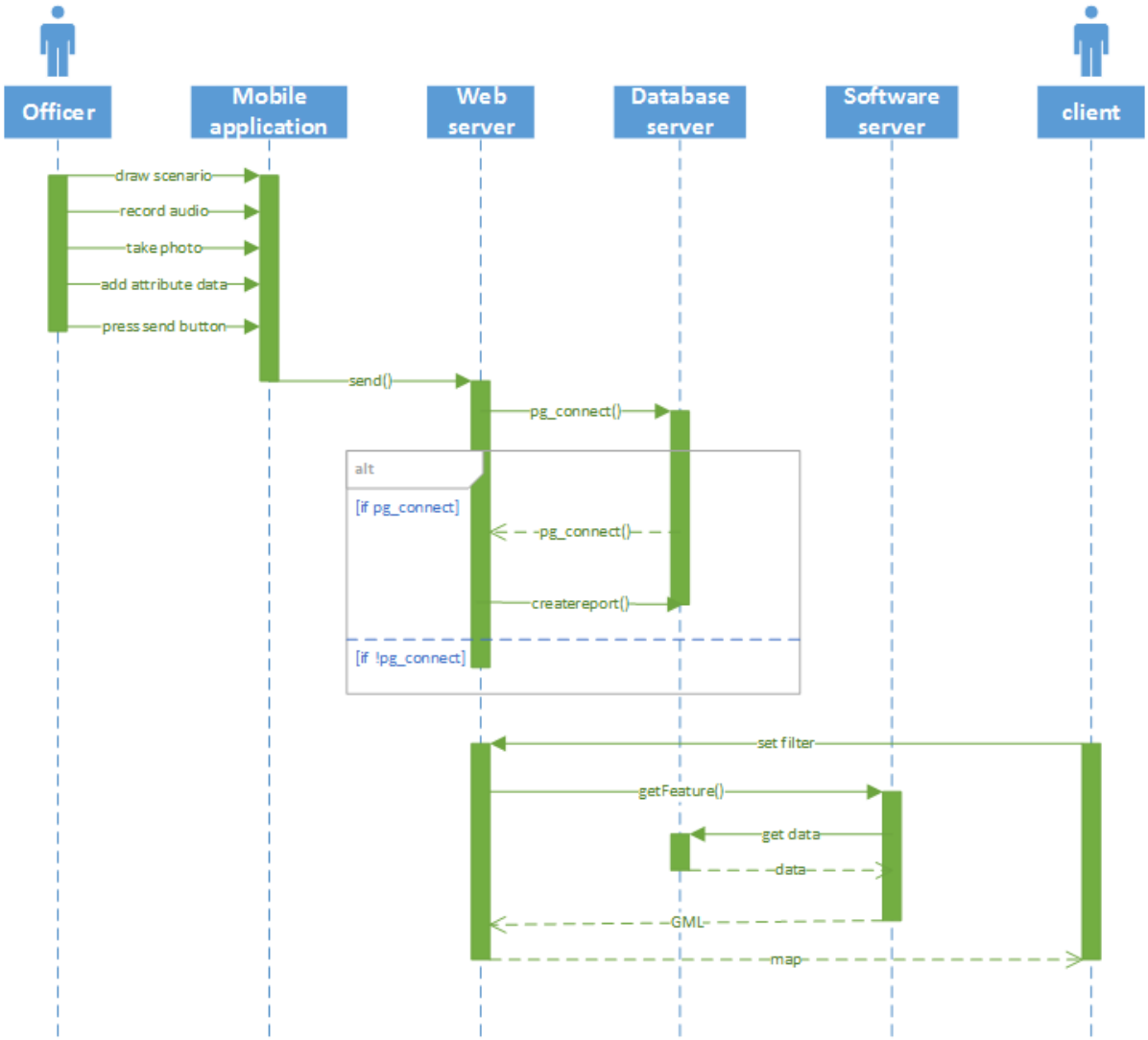


Figure 4.20. UML sequence showing data input and retrieval.

5 Discussion

During the course of the project the Swedish Transport Agency has updated their website with information regarding STRADA. A new web-based client for the digitization by police officers has been launched. A new updated version of the web-based client for extracting reports from STRADA database 2 has been launched as well. However, the need for a mobile application for data collection still exists. The discussions of this chapter only cover the entire system briefly while the mobile application, which is the main contribution to the topic, is discussed in further detail.

5.1 Advantages and implementations

The main advantages of the created mobile application for data collection are that it is structured, easy to use, and offers collection of new media. The mobile application offers input elements such as spinners, or drop down lists, and EditText boxes with filters that control input. Some EditText elements in the application have filters that restrict input to numbers only, others transform input text to uppercase letters. Queries to the PostGIS database are case sensitive, which is why it is important that data that represent the same thing have identical values.

As mentioned in *4.1 Android application*, the layout of a digital form can be kept simple and expand programmatically depending on user input. Functions such as the collection of geographical positions with the help of GPS do not demand any knowledge in the use of a GPS receiver and increase accuracy. The application features some built in help in the form of interactive instructions. The library of symbols of the map tool provides a more structured, and arguably fast, method of visually describing a scene compared to drawing by hand.

Additionally, the mobile application offers collection of new media such as photos, audio, and geographical coordinates. The audio recording feature has been added mainly as an example of possible new features that an application for tablets is capable of providing. The need to record audio has not been mentioned in any of the STRADA documents studied in the project. In fact, it is debatable whether the audio format would be suitable for this kind of application. One can argue that, given the situation, it could be easier to give a detailed explanation of something orally compared to writing it down. On

the other hand, it is not a structured way of storing information. Audio cannot be searched and the attribute data it contains cannot be separated.

5.2 Future works and improvement

An advantage related to future improvement is that a mobile application is easy to upgrade and improve. The goal should be to obtain an application that, with its interactivity, is so easy to use that no manual, such as the one mentioned in *1.2.2 Reporting to STRADA,* is needed. The application could feature more built in help in the form of automatization and constraints. Linking it to databases or featuring more libraries decrease the risk of human errors during data input as well as the amount of time it takes to complete a report. An example is replacing more EditText elements with spinners. Another is using the coordinates of the accident to obtain the name of the municipality or a road. The application has the possibility to feature mandatory input elements and instructions to fill them out following an attempted submission of an incomplete report.

Regarding time efficiency and the issue of late submissions mentioned in *1.2.3 Problems with the current system,* a system with a mobile application for data collection is not only faster (see 1.2.3 Problems with the current system; 5.2.1 More attribute data and connections to other systems) but can be easily modified for further efficiency. Changes to the functionality after software updates should not be a problem for the user if the application has met the goal of having a user interface so easy to understand that no additional tutorial is needed. A built in survey of the user interface could help reaching that goal. Such a survey could measure the duration of the completion of a report with a hidden timer and submit the information along with accident data, although to a separate table. Another example is analysis of the most common mistakes made by users during the use of the application by counting the amount of times error messages and instructions appear.

Regarding improvement of the entire system there is, as mentioned in *3 Methodology and tools*, a need for further work on security. It should be noted that there is no encryption of data, which the current system has (see 1.2.2 Reporting to STRADA), the web server and database do not yet have constraints for e.g. different users and the permitted amount of data requested, and the system lacks backup in the form of mirror databases mentioned in *2 Earlier research, examples and technology*.

The extraction of individual reports from the web service to PDF format in the same style as the downloaded report in Appendix B should be made possible by implementing BIRT, open source software made for this purpose that can be embedded into web applications. With BIRT the developer can create a dynamic template of a report where the data for each field will be retrieved from a connected database (The Eclipse Foundation n.d.-b).

5.2.1 More attribute data and connections to other systems

The paper form has all attribute data attached to the same piece of paper. It requires digitization in order to separate personal and sensitive data from data that will be published in STRADA database number 2 and used by e.g. municipalities. A mobile application eliminates the delay of transporting the document to a police station, forwarding it to the right personnel and digitizing it. The separation is completed early and data are published faster. A system that provides data in nearly real time offers new possibilities of use. Instead of quantitative analysis on fixed occasions, real time warning systems could be developed and used as well. An example is sending relevant data from an accident to specific employees at the municipality in which it occurred or employees at The Swedish Transport Agency working with that region or road. The recipients would ideally have the local knowledge to make connections to, e.g. recent road reconstructions, and act to avoid further accidents. Another example is the attachment of information regarding e.g. damaged road signs. With precise locations and documentation such as photos, municipalities could place orders on new road signs immediately.

5.2.2 Base layers and alternatives to Google Maps

The Google Maps API is currently essential to the mobile application for two reasons: 1. All map related functions are built on Google Maps API, and 2. The base layers come from Google. There were no attempts at alternative solutions, however the implementation of OpenLayers or Leaflet (another map-related JavaScript library like OpenLayers) inside an Android WebView element is a solution that can be examined.

A suitable alternative to the base layers provided by Google is NVDB, an abbreviation of the National Road Database in Swedish. NVDB is already used by STRADA (Sjöö and Ungerbäck 2007) and covers the Swedish road network together

with other data (Swedish Transport Administration 2015). It is the product of collaboration between the Swedish Transport Administration, the Swedish Transport Agency, The Swedish Association of Local Authorities and Regions, forestry and Lantmäteriet (Swedish Transport Administration 2015). The use of NVDB allows linking of attribute data of roads to accidents that occur on them (Sjöö and Ungerbäck 2007) and could increase automatization of the data collection process as proposed in *5.2 Future works and improvement*.

An advantage with owning the base layer is that it allows maximal styling to fit the purpose. A comparison between Figure 1.2 and Figure 4.4 shows a clear difference in proportionality of different map elements, particularly the roads. Roads are visualized narrower in the mobile application whereas the road in the downloaded report in PDF format has a more realistic width. As shown earlier in Figure 4.3, the application does have an option with correct proportions, the layer with satellite imagery, however the imagery contains cars and canopies of trees that at places block the view of the road.

5.2.3 Offline functionality and optimization

The current version of the mobile data collection application relies on internet connectivity for the map, which retrieves its content from Google, and for the submission process (the GPS does not require internet connectivity). With the current extensive coverage of mobile reception, the importance of offline functionality can be argued. However, with occasional dark spots, one would expect a system with offline functionality to run smoother and produce fewer errors.

The design of an offline map with the same functionalities as the current one is something for future commitments. It should be noted that NVDB, as an offline base map, would be an essential resource for such a solution.

Regarding the submission process, the first step to obtaining a design that runs smoother is to assign the task of sending data to the web server to a separate and asynchronous thread in the mobile application. By assigning the operation to a new thread, the responsiveness of the main thread, which contains functionalities of the user interface, will not be affected. This design is especially recommended when the process is a network activity, i.e. it uses internet connectivity (Gargenta 2011). The implementation of separate threads together with a local database could achieve smooth functionality of the application even during loss off internet connectivity. The type of

database in question is SQLite, a single-file serverless database that can be created inside an application (Gargenta 2011; Hipp, Wyrick & Company, Inc. n.d.). SQLite is in the public domain, i.e. it can be used freely for any purpose (Hipp, Wyrick & Company, Inc. n.d.). Figure 5.1 explains the architecture of the proposed solution. A report is saved by storing the collected data locally in the SQLite database. A separate thread is created and attempts to connect to the web server. The thread should contain a loop that puts it to sleep for a period of time if the connection attempt is unsuccessful. When the connection does succeed, the thread sends the data from the local database to the web server for forwarding to the central database. The solution would enable local storage of several reports at the same time.

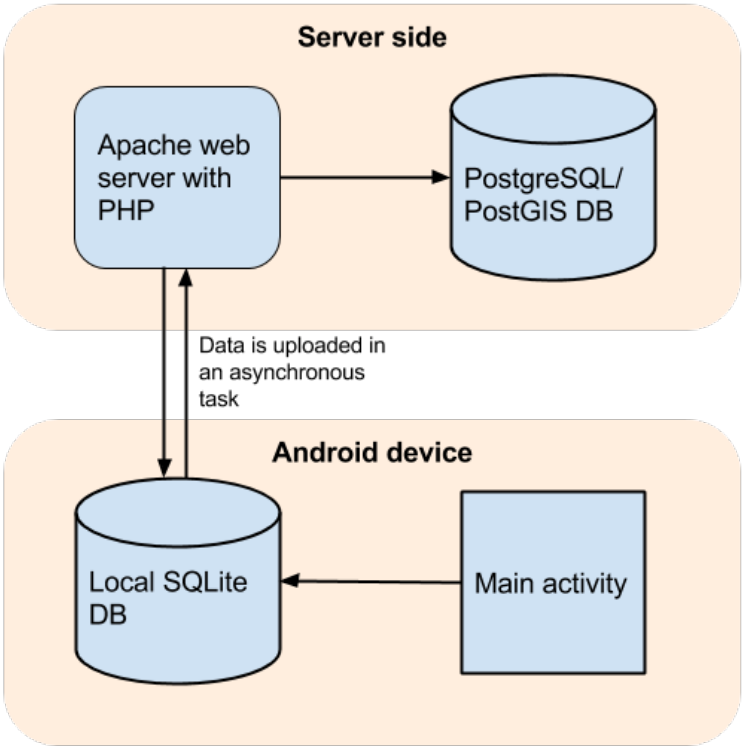


Figure 5.1. Proposed architecture with an SQLite database.

6 Conclusion

The aim of this project is the construction of an application for tablets that could replace the type of form used by police authorities to report traffic accidents; this aim should theoretically be considered as fulfilled. The application can collect the same information as the paper form (see Appendix A) as well as capturing photographs, GPS coordinates, and audio (the last one not mentioned in the aim). The application has a map with the possibility to, by drawing, depict the scene of an accident. As mentioned in *3.6 Google Maps Android API* and *5.2.2 Base layers and alternatives to Google Maps*, the suitability of the base layer provided by Google can be questioned but has not been examined further. The application utilizes other features of a mobile device such as internet connectivity for sending data to a web server.

The implementation of additional components to form an entire system for collection, storage, management, viewing, and sharing of geographic data has led to a deeper understanding of the requirements of output data, and subsequently affected the design of the mobile application. Browsing data in the web service confirms that the mobile application can collect and send data successfully. The functions of the created web service demonstrate that data is stored in suitable formats in a database schema that is flexible enough to facilitate them, i.e. it can be used for the intended purpose: analysis to improve traffic safety.

7 References

Andersson, H., and M. Berg. 2016. ed. Markus. Peter. [E-mail correspondance].

Android Developers. n.d.-a. Android Studio. Retrieved 21 May 2016, from. https://developer.android.com/studio/intro/index.html

Android Developers. n.d.-b. Training. Retrieved 18 May 2016, from. https://developer.android.com/training

Android Developers Blog. 2014. Android Studio 1.0. ed. E. Jamal. http://android-developers.blogspot.se/2014/12/android-studio-10.html.

Android Open Source Project. n.d. Licenses. Retrieved 29 January 2016, from. https://source.android.com/source/licenses.html

Bootstrap. n.d. About Bootstrap. Retrieved 25 May 2016, from. http://getbootstrap.com/about/

Brovelli, A. M., M. Minghini, and G. Zamboni. 2014. Web-based Participatory GIS with data collection on the field - A prototype architecture. *OSGEO Journal,* 13**:** 29-32.

Clegg, P., L. Bruciatelli, F. Domingos, R. R. Jones, M. De Donatis, and R. W. Wilson. 2006. Digital geological mapping with tablet PC and PDA: A comparison. *Computers & Geosciences,* 32**:** 1682-1698. DOI: http://dx.doi.org/10.1016/j.cageo.2006.03.007

DiBona, C., S. Ockman, and M. Stone. 1999. *Open sources : voices from the open source revolution*. Beijing ; Sebastopol, CA: O'Reilly.

El-Gamily, I. H., G. Selim, and E. A. Hermas. 2010. Wireless mobile field-based GIS science and technology for crisis management process: A case study of a fire event, Cairo, Egypt. *The Egyptian Journal of Remote Sensing and Space Science,* 13**:** 21-29. DOI: http://dx.doi.org/10.1016/j.ejrs.2010.07.003

ESRI. n.d. ArcPad. Retrieved 2 September 2016, from. http://www.esri.com/software/arcgis/arcpad

Finn, M. P., and E. L. Usery. 2014. Implications of Web Mercator and Its Use in Online Mapping. *Cartographica: The International Journal for Geographic Information and Geovisualization,* 49**:** 85-101.

Gargenta, M. 2011. *Learning Android*. Sebastopol, Calif.: O'Reilly.

Goodchild, M. F. 2007. Citizens as sensors: the world of volunteered geography. *GeoJournal,* 69**:** 211-221.

Google Developers. 2015. Google Maps/Google Earth APIs Terms of Service: 9. License Requirements. Retrieved 12 February 2016, from. https://developers.google.com/maps/terms - section_9

Hipp, Wyrick & Company, Inc. (Hwaci),. n.d. About SQLite. Retrieved 4 June 2016, from. https://www.sqlite.org/about.html

Ho, D. n.d. About Notepad++. Retrieved 31 January 2016, from. https://notepad-plus-plus.org/

Jenny, B. 2012. Adaptive Composite Map Projections. *IEEE Transactions On Visualization And Computer Graphics,* 18**:** 2575-2582. DOI: 10.1109/TVCG.2012.192

Kresse, W., and D. M. Danko. 2012. *Springer handbook of geographic information*. Berlin: Springer.

Mansourian, A., A. Rajabifard, M. J. Valadan Zoej, and I. Williamson. 2006. Using SDI and web-based system to facilitate disaster management. *Computers and Geosciences,* 32**:** 303-315. DOI: 10.1016/j.cageo.2005.06.017

Markus, P. 2012. Kartläggning av cykelolyckor i Malmö (Mapping of Bicycle Accidents in Malmö). Malmö Högskola.

Mattsson, K., and A. Ungerbäck. 2013. *Vägtrafikolyckor : handledning vid rapportering* (Traffic Accidents : Tutorial for Reporting). Borlänge: Transportstyrelsen.

Merriam-Webster. n.d. Application Programming Interface. Retrieved 8 April 2016, from. http://www.merriam-webster.com/dictionary/application programming interface

Mozilla. n.d. What is Firebug. Retrieved 31 January 2016, from. http://getfirebug.com/whatisfirebug

Näringsdepartementet RS T. 1986. Kungörelse om statistiska uppgifter angående vägtrafikolyckor (Proclamation of Statistical Data Concerning Traffic Accidents). In *SFS 1965:561,* ed. Regeringskansliet. Svensk Författningssamling.

O'Reilly, T. (2005) What is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. http://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html (last accessed 22 May 2016).

Open Geospatial Consortium. n.d.-a. Retrieved 18 February 2016, from. http://www.opengeospatial.org/

Open Geospatial Consortium. n.d.-b. Members. Retrieved 18 February 2016, from. http://www.opengeospatial.org/ogc/members

Open Geospatial Consortium. n.d.-c. Web Map Service. Retrieved 1 November 2016, from. http://www.opengeospatial.org/standards/wms

Open Handset Alliance. 2007. FAQ. Retrieved 20 May 2016, from. http://www.openhandsetalliance.com/oha_faq.html

Open Handset Alliance. n.d. Members. Retrieved 20 May 2016, from. http://www.openhandsetalliance.com/oha_members.html

GeoServer. 2.7.1, Open Source Geospatial Foundation,, http://geoserver.org/release/2.7.1/.

Open Source Geospatial Foundation. n.d.-a. GeoServer User Manual. Retrieved 19 November 2015, from. http://docs.geoserver.org/2.7.1/user/index.html

Open Source Geospatial Foundation. n.d.-b. GeoServer User Manual, CQL and ECQL. Retrieved 24 April 2016, from. http://docs.geoserver.org/2.7.1/user/tutorials/cql/cql_tutorial.html - cql-tutorial

Open Source Geospatial Foundation. n.d.-c. GeoServer User Manual, History. Retrieved 16 February 2016, from. http://docs.geoserver.org/2.7.1/user/introduction/history.html

Open Source Geospatial Foundation. n.d.-d. GeoServer User Manual, Web archive. Retrieved 19 November 2015, from. http://docs.geoserver.org/2.7.1/user/installation/war.html

Open Source Geospatial Foundation. n.d.-e. GeoServer User Manual, WFS reference. Retrieved 22 May 2016, from. http://docs.geoserver.org/2.7.x/en/user/services/wfs/reference.html

Open Source Geospatial Foundation. n.d.-f. GeoServer User Manual, WMS reference. Retrieved 22 May 2016, from. http://docs.geoserver.org/2.7.x/en/user/services/wms/reference.html

Open Source Geospatial Foundation. n.d.-g. What is GeoServer? Retrieved 19 November 2015, from. http://geoserver.org/about/

Open Source Initiative. 2012. History. Retrieved 21 April 2016, from. https://opensource.org/history

Open Source Initiative. n.d.-a. The BSD 2-Clause License. Retrieved 1 June 2015, from. https://opensource.org/licenses/BSD-2-Clause

Open Source Initiative. n.d.-b. GNU General Public License, version 2 (GPL-2.0). Retrieved 12 November 2015, from. http://opensource.org/licenses/gpl-2.0.php
Open Source Initiative. n.d.-c. Licenses & Standards. Retrieved 22 April 2016, from. https://opensource.org/licenses
Open Source Initiative. n.d.-d. The MIT license (MIT). Retrieved 25 May 2016, from. https://opensource.org/licenses/MIT
Open Source Initiative. n.d.-e. Open Source Definition (Annotated). Retrieved 21 April 2016, from. https://opensource.org/osd-annotated
Open Source Initiative. n.d.-f. The PostgreSQL License (PostgreSQL). Retrieved 12 November 2015, from. http://opensource.org/licenses/postgresql
OpenLayers 3. n.d. Retrieved 1 June 2016, from. http://openlayers.org/
Oracle. n.d. Java Servlet Technology. Retrieved 2 March 2016, from. http://www.oracle.com/technetwork/java/index-jsp-135475.html
Persson, G., and I. Uusmann. 1997. Regeringens proposition 1996/97:137, Nollvisionen och det trafiksäkra samhället (The Vision of Zero Fatalities and a Society with Safe Traffic). Regeringen.
pgAdmin. n.d. Retrieved 19 November 2015, from. http://www.pgadmin.org/
Refractions Research. n.d. What is PostGIS? Retrieved 12 November 2015, from. http://postgis.refractions.net/
Sjöö, B., and A.-C. Ungerbäck. 2007. *Nytt nationellt informationssystem för skador och olyckor inom hela vägtransportsystemet : Strada : slutrapport* (New National Information System for Injuries and Accidents in the Entire Road Transport System: Strada: Final Report). Borlänge: Vägverket.
Statista. n.d. Number of apps available in leading app stores as of July 2015. Retrieved 27 April 2016, from. http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/
Sublime Text. n.d. Sublime Text. Retrieved 31 January 2016, from. http://www.sublimetext.com/
Swedish Transport Administration. 2015. About NVDB. Retrieved 2 June 2016, from. http://www.nvdb.se/en/om-nvdb/
Swedish Transport Agency. n.d.-a. Bakgrund. Retrieved 11 September 2015, from. https://www.transportstyrelsen.se/sv/vagtrafik/statistik-och-register/STRADA-informationssystem-for-olyckor-skador/Bakgrund/
Swedish Transport Agency. n.d.-b. Rapportörer och användare av statistik. Retrieved 26 January 2015, from. https://www.transportstyrelsen.se/sv/Vag/STRADA-informationssystem-for-olyckor-skador/Rapportorer-och-anvandare/
Swedish Transport Agency. n.d.-c. Strada – informationssystem för olyckor och skador i trafiken. Retrieved 26 January 2015, from. https://www.transportstyrelsen.se/sv/Vag/STRADA-informationssystem-for-olyckor-skador/
Tallungs, S., and A. Josefsson (2014) Haveriet inifrån: Så gick Pust från succé till fiasko (From Inside the Failure: How Pust Went from Success to Fiasco). Debate article co-written by the implementation manager and the usability manager. http://computersweden.idg.se/2.2683/1.659798/agilt-15-tecken (last accessed 23 April 2015).
Tao, J. 2010. Exploring Massive Volunteered Geographic Information for Geographic Knowledge Discovery. PhD Thesis KTH Royal Institute of Technology
The Apache Software Foundation. 2004. Apache License Version 2.0. Retrieved 31 May 2016, from. http://www.apache.org/licenses/LICENSE-2.0

The Apache Software Foundation. n.d. Apache Tomcat. Retrieved 17 August 2015, from. http://tomcat.apache.org
The Eclipse Foundation. n.d.-a. Retrieved 22 May 2016, from. https://eclipse.org
The Eclipse Foundation. n.d.-b. About BIRT. Retrieved 1 November 2016, from. http://www.eclipse.org/birt/about/
The jQuery Foundation. n.d. License. Retrieved 30 May 2016, from. https://jquery.org/license/
The Linux Foundation. n.d. What is Linux? Retrieved 21 May 2016, from. https://www.linux.com/what-is-linux
The Mozilla Foundation. n.d. HTTP access control (CORS). Retrieved 2 September 2016, from. https://developer.mozilla.org/en-US/docs/Web/HTTP/Access_control_CORS
The PHP Group. n.d.-a. What can PHP do? Retrieved 27 April 2016, from. http://php.net/manual/en/intro-whatcando.php
The PHP Group. n.d.-b. What is PHP? Retrieved 27 April 2016, from. http://php.net/manual/en/intro-whatis.php
The PostgreSQL Global Development Group. n.d.-a. About JDBC. Retrieved 21 May 2016, from. https://jdbc.postgresql.org/about/about.html
The PostgreSQL Global Development Group. n.d.-b. About PostgreSQL. Retrieved 12 November 2015, from. http://www.postgresql.org/about/
Ungerbäck, A. 24 April 2015. Swedish Transport Agency. [E-mail correspondance].
Ungerbäck, A. 2008. Projektspecifikation för utveckling av polisrapporter till STRADA (Project specification for development of digital transfer of police reports to STRADA). [Project specification].
Van Kasteren, A. 2014. Cross-Origin Resource Sharing. Retrieved 2 September 2016, from. https://www.w3.org/TR/cors/
Vägverket. 1996. Vägtrafikens skade- och olycksstatistik : regeringsuppdrag (Road Traffic Injury and Accident Statistics: Government Mission). In *Publikation / Vägverket, 99-0651839-1 ; 1996:43*. Borlänge: Vägverket.

Appendices

Appendix A

The attachment is the form that is currently used by police in situ for collecting information about traffic accidents.

Source:

Mattsson, K., and A. Ungerbäck. 2013. *Vägtrafikolyckor : handledning vid rapportering* (Traffic Accidents : Tutorial for Reporting). Borlänge: Transportstyrelsen.

51 Polismyndigheten (arbetsenhet, telefon)

INFORMATIONSUNDERLAG 52 Polisens diarienummer

Vägtrafikolycka

A

53 Polismynd.kod	54 Kommun	55 Tidpunkt för olyckan	År	Mån	Dag	Kl	Veckodag	Väghållarkod

56 Olycksplats (ange gatu-/vägnamn/vägnr, ev husnr samt avstånd till närmaste korsning mellan allmänna vägar)

Namn på stadsdel/kommundel/ort el dyl

57 Skiss, på vilken anges gatu- och vägnamn, vägbredd, åtföljd av bokstav A resp B enl. avsnitt B nedan. Vid inritat fordon anges fordonsslag (pb, lb, etc) ett trafikelement - (vägtrafikant-) nummer 1, 2, 3 osv, vilket nr skall vara identiskt med det nr vederbörande vägtrafikant åsatts i trafikmålsanteckningar (RPS 411.20)

Norrpil

58 Kortfattad beskrivning av händelseförloppet, siktförhållanden m.m.

B Väg- och Trafik

	Väg A	Väg B
59 Vägnummer		
60 Högsta tillåtna hastighet		
61 Vägtyp		
Motorväg 1		
Motortrafikled 2		
Annan allm väg 3		
Gata 4		
Enskild väg 5		
Övr väg, torg etc 6		

62 **Trafikanvisningar*)**	Väg A	Väg B
Huvudled 1		
Ej huvudled 2		
63 Trafikreglering*)		
Förb mot v-sväng 1		
Stopplikt 2		
Väjningsplikt 3		
64 Trafiksignal*)		
I funktion 1		
Ur funktion 2		
Gult blinkande 3		
Saknas 4		

C Väderlek, väglag, belysning

65 Väderleksförhållanden	
Uppehållsväder 1	
Dis/dimma 2	
Regn 3	
Snöblandat regn 4	
Snöfall 5	
66 Väglag	
Vägbanan torr 1	
Vägbanan våt/fuktig 2	
Tjock is/packad snö 3	
Tunn is (vägb synlig) 4	
Lös snö/snömodd 5	

67 Trafikmiljö	
Tättbebyggt område 1	
Ej tättbebyggt område 2	
68 Ljusförhållanden	
Dagsljus 1	
Mörker 2	
Gryning/skymning 3	
Om 68:2 eller 3 förkryssats	

69 Gatu-/vägbelysning	Väg A	Väg B
Tänd 1		
Släckt 2		
Saknas 3		

D Trafikelement

E Inblandade personer

70 Trafikelement						71 Personnummer	72 Trafikant				73 Personskada			
Nr	Trafikelement (t.ex. pb, lätt/tung lb, lätt/tung mc, cykel, gående enl. 1 Kap. 4 § TrF, vilt/djur)	Registreringsnr. (anges för motor- och släpfordon). För utländskt fordon, nationalitet	Totalt antal pers i fordonet	Övningskörning**) Trafik skola	Privat	Obligatoriskt för förare och instruktör samt dödade och skadade personer	Förare el. elev som kör. Ange F/E	Passagerare/ instruktör Fram	Bak	Okänt eller övrigt	Död	Svårt skadad	Lindrigt skadad	74 Misstänkt påverkad av alkohol/annat ämne (förare). Ange J/N

Fordon skyltat för transport av farligt gods inblandat. Ange elementnr:

FU inledd, datum och tid | av (titel, för- och efternamn)

Ort och datum

Uppgiftslämnare

50 Statistiska uppgifter till Vägverket | Datum och sign

75 Undersökn.ledarens beslut

- FU inleds ej
- Ej spaningsresultat
- Brott kan ej styrkas
- Misstänkt oskyldig
- Gärningen ej brott
- FU nedlagd
- Misstänkt ej fyllt 15 år
- Misstänkt avliden
- Rapporteftergift
- Spaningsuppslag saknas

Beslutsdatum

Undersökningsledarens namnteckning/sign

*) Kontrolleras **) Med övningskörning avses enbart de fall då eleven framfört fordonet, alltså ej då instruktören kört.

Undersökningsledarens beslut

Polismyndigheten / kopia till Vv / försäkringsbolag

Appendix B

The attachment is an example of a digitized, submitted and downloaded report in PDF-format.

Source:

Swedish Transport Agency, Exempel på statistikuttag, Retrieved 26 Jan 2015, from. https://www.transportstyrelsen.se/sv/vagtrafik/statistik-och-register/STRADA-informationssystem-for-olyckor-skador/Exempel-pa-uttag/

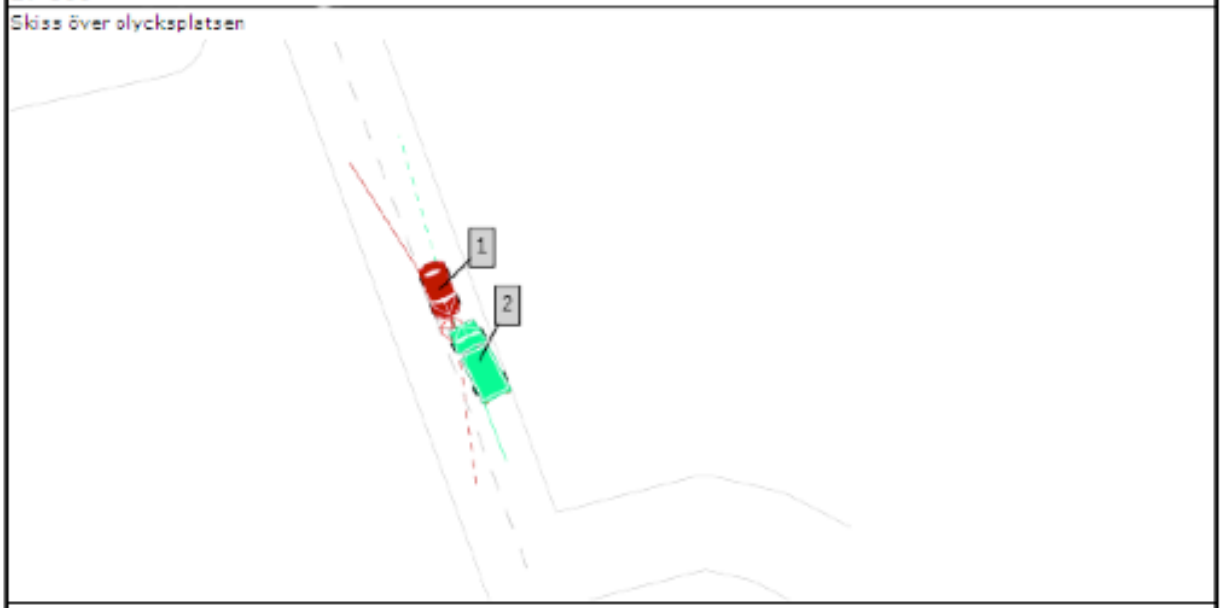
Polisrapport Vägtrafikolycka

Olycks-ID	Polisens diarienummer
9067	T-344 /10

Län	Kommun	Tidpunkt för olyckan	Olyckstyp
Stockholms län	Ekerö	2010-XX-XX	M - Möte (motorfordon)

Olycksplats
Lv 800

Skiss över olycksplatsen



Beskrivning av händelseförloppet
Pb 1 har kommit över i mötande körfält och kolliderat med mötande tung lb 1.

Väderleksförhållanden	Väglag	Trafikmiljö
Uppehållsväder	Vägbanan torr	Ej tättbebyggt område
Ljusförhållanden	**Platstyp**	**Attribut**
Dagsljus	Gatu-/Vägsträcka	

	Väg A	Väg B
Vägnummer/Gatunamn	Färentunavägen, 800	
Högsta tillåtna hastighet	70 km/h	
Vägtyp	Annan allmän väg	
Trafikanvisning	Okänt	
Trafikreglering		
Trafiksignal	Okänt	
Gatu-/vägbelysning	Uppgift saknas	

Trafikelement

Nr	Typ	Totalt antal personer	Övnings-körning
1	Personbil	1	
2	Lastbil (tung)	2	

Trafikant

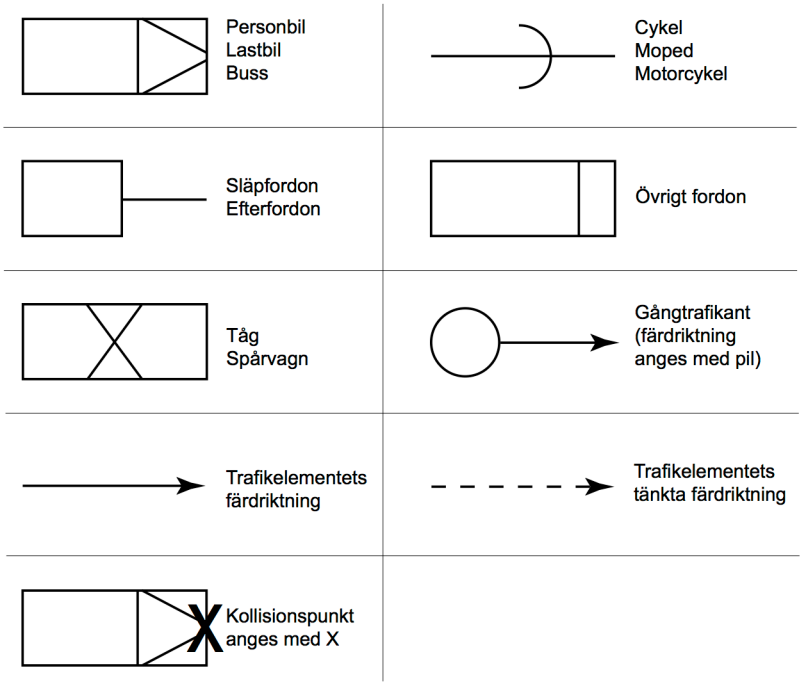
Nr	Ålder och kön	Förare	Passagerare			Personskada				Föraren
			Fram	Bak	Okänt	Död	Svår	Lindrig	Okänd	Misstänkt påverkad
1	XX-M	X					X			O
2	XX-M	X								N

Appendix C

This is a figure from the tutorial for filling out the form attached in Appendix A. The figure describes the symbology that the police officer should use when drawing the scenario by hand. The symbols represent, from left to right, first row: a car, truck or bus, a bicycle, moped or motorcycle, second row: a trailer, other vehicle types, third row: a train- or tramcar, a pedestrian (the direction of travel is marked with arrow), fourth row: direction of travel and intended direction of travel of the traffic element, fifth row: point of collision (marked with ”X”).

Source:

Mattsson, K., and A. Ungerbäck. 2013. *Vägtrafikolyckor : handledning vid rapportering* (Traffic Accidents : Tutorial for Reporting). Borlänge: Transportstyrelsen.



Institutionen för naturgeografi och ekosystemvetenskap, Lunds Universitet.
Student examensarbete (Seminarieuppsatser). Uppsatserna finns tillgängliga på institutionens geobibliotek, Sölvegatan 12, 223 62 LUND. Serien startade 1985. Hela listan och själva uppsatserna är även tillgängliga på LUP student papers (https://lup.lub.lu.se/student-papers/search/) och via Geobiblioteket (www.geobib.lu.se)
The student thesis reports are available at the Geo-Library, Department of Physical Geography and Ecosystem Science, University of Lund, Sölvegatan 12, S-223 62 Lund, Sweden. Report series started 1985. The complete list and electronic versions are also electronic available at the LUP student papers (https://lup.lub.lu.se/student-papers/search/) and through the Geo-library (www.geobib.lu.se)

350 Mihaela – Mariana Tudoran (2015) Occurrences of insect outbreaks in Sweden in relation to climatic parameters since 1850
351 Maria Gatzouras (2015) Assessment of trampling impact in Icelandic natural areas in experimental plots with focus on image analysis of digital photographs
352 Gustav Wallner (2015) Estimating and evaluating GPP in the Sahel using MSG/SEVIRI and MODIS satellite data
353 Luisa Teixeira (2015) Exploring the relationships between biodiversity and benthic habitat in the Primeiras and Segundas Protected Area, Mozambique
354 Iris Behrens & Linn Gardell (2015) Water quality in Apac-, Mbale- & Lira district, Uganda - A field study evaluating problems and suitable solutions
355 Viktoria Björklund (2015) Water quality in rivers affected by urbanization: A Case Study in Minas Gerais, Brazil
356 Tara Mellquist (2015) Hållbar dagvattenhantering i Stockholms stad - En riskhanteringsanalys med avseende på långsiktig hållbarhet av Stockholms stads
dagvattenhantering i urban miljö
357 Jenny Hansson (2015) Trafikrelaterade luftföroreningar vid förskolor – En studie om kvävedioxidhalter vid förskolor i Malmö
358 Laura Reinelt (2015) Modelling vegetation dynamics and carbon fluxes in a high Arctic mire
359 Emelie Linnéa Graham (2015) Atmospheric reactivity of cyclic ethers of relevance to biofuel combustion
360 Filippo Gualla (2015) Sun position and PV panels: a model to determine the best orientation
361 Joakim Lindberg (2015) Locating potential flood areas in an urban environment using remote sensing and GIS, case study Lund, Sweden
362 Georgios-Konstantinos Lagkas (2015) Analysis of NDVI variation and snowmelt around Zackenberg station, Greenland with comparison of ground data and remote sensing.
363 Carlos Arellano (2015) Production and Biodegradability of Dissolved Organic Carbon from Different Litter Sources
364 Sofia Valentin (2015) Do-It-Yourself Helium Balloon Aerial Photography - Developing a method in an agroforestry plantation, Lao PDR
365 Shirin Danehpash (2015) Evaluation of Standards and Techniques for Retrieval of Geospatial Raster Data - A study for the ICOS Carbon Portal
366 Linnea Jonsson (2015) Evaluation of pixel based and object based classification methods for land cover mapping with high spatial resolution satellite imagery, in the Amazonas, Brazil.
367 Johan Westin (2015) Quantification of a continuous-cover forest in Sweden

using remote sensing techniques

368 Dahlia Mudzaffar Ali (2015) Quantifying Terrain Factor Using GIS Applications for
Real Estate Property Valuation

369 Ulrika Belsing (2015) The survival of moth larvae feeding on different plant species in northern Fennoscandia

370 Isabella Grönfeldt (2015) Snow and sea ice temperature profiles from satellite data and ice mass balance buoys

371 Karolina D. Pantazatou (2015) Issues of Geographic Context Variable Calculation Methods applied at different Geographic Levels in Spatial Historical Demographic Research -A case study over four parishes in Southern Sweden

372 Andreas Dahlbom (2016) The impact of permafrost degradation on methane fluxes - a field study in Abisko

373 Hanna Modin (2016) Higher temperatures increase nutrient availability in the High Arctic, causing elevated competitive pressure and a decline in *Papaver radicatum*

374 Elsa Lindevall (2016) Assessment of the relationship between the Photochemical Reflectance Index and Light Use Efficiency: A study of its seasonal and diurnal variation in a sub-arctic birch forest, Abisko, Sweden

375 Henrik Hagelin and Matthieu Cluzel (2016) Applying FARSITE and Prometheus on the Västmanland Fire, Sweden (2014): Fire Growth Simulation as a Measure Against Forest Fire Spread – A Model Suitability Study –

376 Pontus Cederholm (2016) Californian Drought: The Processes and Factors Controlling the 2011-2016 Drought and Winter Precipitation in California

377 Johannes Loer (2016) Modelling nitrogen balance in two Southern Swedish spruce plantations

378 Hanna Angel (2016) Water and carbon footprints of mining and producing Cu, Mg and Zn: A comparative study of primary and secondary sources

379 Gusten Brodin (2016) Organic farming's role in adaptation to and mitigation of climate change - an overview of ecological resilience and a model case study

380 Veränika Trollblad (2016) Odling av *Cucumis Sativus* L. med aska från träd som näringstillägg i ett urinbaserat hydroponiskt system

381 Susanne De Bourg (2016) Tillväxteffekter för andra generationens granskog efter tidigare genomförd kalkning

382 Katarina Crafoord (2016) Placering av energiskog i Sverige - en GIS analys

383 Simon Nåfält (2016) Assessing avalanche risk by terrain analysis An experimental GIS-approach to The Avalanche Terrain Exposure Scale (ATES)

384 Vide Hellgren (2016) Asteroid Mining - A Review of Methods and Aspects

385 Tina Truedsson (2016) Hur påverkar snömängd och vindförhållande vattentrycksmätningar vintertid i en sjö på västra Grönland?

386 Chloe Näslund (2016) Prompt Pediatric Care Pediatric patients' estimated travel times to surgically-equipped hospitals in Sweden's Scania County

387 Yufei Wei (2016) Developing a web-based system to visualize vegetation trends by a nonlinear regression algorithm

388 Greta Wistrand (2016) Investigating the potential of object-based image analysis to identify tree avenues in high resolution aerial imagery and lidar data

389 Jessica Ahlgren (2016) Development of a Web Mapping Application for

grazing resource information in Kordofan, Sudan, by downloading MODIS data automatically via Python

390 Hanna Axén (2016) Methane flux measurements with low-cost solid state sensors in Kobbefjord, West Greenland

391 Ludvig Forslund (2016) Development of methods for flood analysis and response in a Web-GIS for disaster management

392 Shuzhi Dong (2016) Comparisons between different multi-criteria decision analysis techniques for disease susceptibility mapping

393 Thirze Hermans (2016) Modelling grain surplus/deficit in Cameroon for 2030

394 Stefanos Georganos (2016) Exploring the spatial relationship between NDVI and rainfall in the semi-arid Sahel using geographically weighted regression

395 Julia Kelly (2016) Physiological responses to drought in healthy and stressed trees: a comparison of four species in Oregon, USA

396 Antonín Kusbach (2016) Analysis of Arctic peak-season carbon flux estimations based on four MODIS vegetation products

397 Luana Andreea Simion (2016) Conservation assessments of Văcărești urban wetland in Bucharest (Romania): Land cover and climate changes from 2000 to 2015

398 Elsa Nordén (2016) Comparison between three landscape analysis tools to aid conservation efforts

399 Tudor Buhalău (2016) Detecting clear-cut deforestation using Landsat data: A time series analysis of remote sensing data in Covasna County, Romania between 2005 and 2015

400 Sofia Sjögren (2016) Effective methods for prediction and visualization of contaminated soil volumes in 3D with GIS

401 Jayan Wijesingha (2016) Geometric quality assessment of multi-rotor unmanned aerial vehicle-borne remote sensing products for precision agriculture

402 Jenny Ahlstrand (2016) Effects of altered precipitation regimes on bryophyte carbon dynamics in a Peruvian tropical montane cloud forest

403 Peter Markus (2016) Design and development of a prototype mobile geographic information system for real-time collection and storage of traffic accident data