

A Community Based Approach to Traffic Information Systems

Master Thesis in Computer Science

Mats Lindh

October 28, 2005
Halden, Norway



Høgskolen i Østfold
Avdeling for Informasjonsteknologi

Abstract

This thesis describes a traffic information system based on community participation. Each member of the community can submit, edit and delete information in the repository. The system is based on the wiki approach to content creation, and shows that the spatial and temporal qualities of traffic information makes it ideal for community processing. The mobile clients are based on available consumer grade hardware, and their position is determined by a Bluetooth GPS. Information reported by the clients are used to generate background imaging of road networks, and the server maintains a historical repository of data. The mobile clients retrieve and report data using Python for Series 60, and the communication is implemented through XML-RPC. A dynamic Scalable Vector Graphics (SVG) document is used for desktop presentation, and traffic information is distributed as a georeferenced Really Simple Syndication (RSS 2.0) stream.

Acknowledgements

I thank Anette for her continuous support through the inspired and uninspired moments of this thesis, it would never have been possible without you. I thank my family for their support, and Gunnar Misund for his supervision with the thesis and for numerous suggestions and ideas. I also thank Kristian, Bjørn Håkon, Linda and Hilde, who have been writing their theses on the computer lab this semester, for making the process less tiresome.

I am grateful for the proofreading performed by Christer Edvartsen, Anders Peter Harriðsleff, Eirik Jensen, Jørn-Inge Kristiansen and in particular Eirik Refsdal. This thesis would not have reached its current state without your suggestions and support.

NRK and P4 was very helpful with providing background information about current traffic systems, and helped the progress of this thesis.

Contents

Abstract	i
Acknowledgements	iii
1 Introduction	1
2 Background	5
2.1 Mobile Computing	5
2.1.1 Mobile Networks	5
2.1.2 Mobile Application Development	7
2.2 Open Geospatial Specifications and Services	11
2.2.1 Geography Markup Language (GML)	12
2.2.2 Web Map Service (WMS)	13
2.2.3 GPS Exchange Format (GPX)	15
2.2.4 The NMEA GPS Protocol	16
2.3 Location Based Services	16
2.3.1 What is a Location Based Service?	17
2.3.2 Introduction to Mobile Positional Methods	17
2.3.3 OpenGIS Location Services (OpenLS)	19
2.4 User Generated Content	20
2.4.1 Blogs	20
2.4.2 Wikis	21
2.4.3 Citizen Journalism	22
2.4.4 Metadata	23
2.5 Traffic Information and Car Navigational Systems	24
2.5.1 Extracts from Car Navigation Systems Research	25
2.5.2 Current Industry Standards	26
2.5.3 Current Industry Practice	26
2.5.4 Available Commercial Car Navigation Solutions	29
2.6 Other Standards of Interest	32
2.6.1 XML-RPC	32
2.6.2 Scalable Vector Graphics (SVG)	33
2.6.3 Really Simple Syndication (RSS)	33

2.7	Conclusions	33
3	Scenarios	35
3.1	Informed About Traffic Conditions	35
3.2	Manually Reporting Traffic Conditions	35
3.3	Annotating Traffic Information with Road Maps	36
3.4	Dynamic Maps - Mobile Device	37
3.5	Dynamic Maps - Desktop Computer, Route Planning	37
4	System Design	39
4.1	System and Problem Description	39
4.1.1	Obtaining Traffic and Event Information	39
4.1.2	Automatic Retrieval of Traffic Information	40
4.1.3	Validating Traffic and Event Information	40
4.1.4	Distributing Traffic and Event Information	41
4.1.5	Presenting Traffic and Event Information	41
4.2	Mobile Devices	42
4.2.1	Application Development	42
4.3	Positioning	43
4.4	Communication	44
4.5	Storage and Transfer Formats	45
4.6	Presentation	46
5	Prototype Implementation	49
5.1	Implementation Overview	49
5.2	Mobile Device Client	50
5.3	Traffic Server	51
5.4	Presentation	53
5.4.1	SVG Interface	54
5.4.2	RSS Interface	56
5.5	NMEAReplayer	58
6	System Tests	64
6.1	User Case Test	64
6.2	Performance Tests	66
7	Discussion and Future Work	71
7.1	Discussion	71
7.2	Future Work	72
7.3	In summary	74
8	Conclusions	75

A	Examples	77
A.1	NMEA Log Example	77
A.1.1	Extract from a NMEA log file	77
A.2	RSS Feed Examples	78
A.2.1	Georeferenced RSS Feed Showing The Three Categories of Traffic Slow-downs	78
A.3	SVG Basic Examples	79
A.3.1	Red Circle	79
A.4	XML-RPC Request and Response Example	79
A.4.1	Request	79
A.4.2	Response	80
B	External Documentation	81
B.1	Traffic Server	81
	Bibliography	82

List of Figures

2.1	The merge of different mobile devices	6
2.2	UMTS coverage map for southern Østfold, Norway, 22. May 2005	7
2.3	Java MIDP2 and some of its optional modules	8
2.4	A comparison between the Java MIDP2 and the Symbian OS.	9
2.5	GML extended with metadata information in the Dublin Core grammar	12
2.6	WMS server structure	13
2.7	WMS layer merging	14
2.8	WMS server aggregation	15
2.9	WMS GetFeatureInfo functionality	15
2.10	GPX used as a transport format	16
2.11	Determining position by cell tower location	17
2.12	Mobile smartphones and Bluetooth equipment	18
2.13	OpenLS information model (from [1])	19
2.14	The inherent network structure between blogs	21
2.15	Editing a page on Wikipedia	22
2.16	Screenshot from Verdens Gang (VG)	23
2.17	Screenshot from the Heimdall application	27
2.18	Screenshot from the Heimdall application	28
2.19	Screenshot from the Heimdall application	29
2.20	Screenshot from the Heimdall application	30
2.21	The flow of an invocation by XML-RPC (from [2])	33
3.1	Scenario 3.3: Christer receives accident information	36
3.2	Scenario 3.5: A road section is closed, and a client chooses another road	38
4.1	The lifetime and states of a message	41
4.2	XML-RPC and the advantage of using HTTP as a transport protocol	45
4.3	The presentation layer provides several output formats to clients	47
4.4	RSS and SVG representations of data	47
5.1	Architectural overview	49
5.2	The Client: Adding a new event in the field	51
5.3	The Client: Deleting the new event	52

5.4	Traffic Server XML-RPC log	53
5.5	Grid index id is determined by snapping the location to the grid, and selecting the correct section for the heading	54
5.6	SVG Interface: Viewing an event and its details	55
5.7	SVG Interface: Viewing an event and one active client in the map	56
5.8	SVG Interface: Viewing several events and active clients in the map	57
5.9	SVG Interface: Flow schematic for dynamic SVG	58
5.10	SVG Interface: Generated background map from client history	59
5.11	SVG Interface: Generated local background map from client history	60
5.12	SVG Interface: Another generated local background map from client history	61
5.13	SVG Interface: Road segments weighted by velocity	62
5.14	RSS Interface: Displaying the RSS stream and its messages in Mozilla Thunderbird	63
5.15	Screenshot from NMEAReplayer	63
6.1	The two paths through Fredrikstad city center	65
6.2	Number of cars arriving at intersection divided into five minute intervals	66
6.3	Transcript of an XML-RPC session from ethereal	67
6.4	Ethereal statistics from a complete 35 minute long session at one second intervals	68
6.5	Comparison of quality relating to size of update intervals	69

List of Tables

6.1	Travel time relative to traffic conditions	65
6.2	Average response times from the Traffic Server depending on network connection .	66
6.3	Size of transmitted data in relation to update interval	68

Chapter 1

Introduction

This master thesis describes a traffic information system based on community effort and information sharing. I describe a solution that by using consumer grade hardware available on the market, allows the public to participate in the process. The traffic situation is an interaction between objects such as vehicles and pedestrians, and to discover and avoid negative traffic situations, an observation must exist and be distributed to other people [3]. Information is currently distributed through traditional media, although some research and commercial projects have experimented with digital distribution [4] [5]. The current situation is that most traffic information is distributed as audible messages through radio programming, and is unavailable for aggregation or computational processing. Car navigation systems are unable to use this information, and situations that only exist in a limited time are never reported.

The system described in this thesis uses a digital solution, where content collection, generation and distribution is performed in a client/server setting. This allows for the distribution of content relating to the location of the recipient. I also explore the idea of automatically detecting traffic related problems in an area, by examining the position and speed of a vehicle in comparison to the expected values at a location. If a major deviation is present, an automatic event is triggered. Event information is distributed to clients within a range of the event location, or have requested information about the area where the event occurred. Information is provided by several methods where the user can choose the solution that matches his need.

The reported events are available for editing by the community. Individuals can update, delete and add information without the approval of an authority. The changes can be returned to the previous state by any other member of the community. The idea is that people participate for the common good of the community, and the community is able to correct erroneous information. Different presentation technologies are explored, for presentation to desktop clients and exporting data to other applications through resource frameworks such as Really Simple Syndication (RSS) [6].

This idea is based on the concepts of user created and user contributed information. The recent change on the Internet towards wiki [7] based solutions inspired this work, and is an example that such projects can be successful. Research indicates the required levels of information reliability [8]. In addition I wanted to briefly explore the area of automatic detection of traffic related information, and to detect problems that do not warrant manual interaction. Potential applications for the project include commercial traffic solutions, tools for traffic analysis, distributed architectures and road

usage information. The statistical properties of the data can also be extracted, together with path and topological information.

The rest of this introduction contains a short summary of each of the upcoming chapters, providing a overview of their contents.

Chapter 2 - Background

The Background chapter introduces the research domains of this thesis. These include mobile devices and available network types, including an introduction the available development environments. Section 2.1, open geospatial standards, introduce the Web Map Service, the Web Feature Service and the Web Coverage Service. The description of Location Based Services builds on the ideas from the two previous sections, and introduces the OpenLS specification. Using user created content in an application is introduced in Section 2.4, while Section 2.5 presents research and commercial products relating to traffic information systems. The chapter then leads into introductions to Scalable Vector Graphics (SVG) and XML-RPC, before ending with a summarization and a short evaluation of the described technologies.

Chapter 3 - Scenarios

A collection of five scenarios describing the functionality of the system. The scenarios are referenced in Chapter 4, where solutions to obtain the functionality is presented. The scenarios include mobile clients, desktop presentation and automatically detecting changes in the traffic flow.

Chapter 4 - System Design

A description of the system design of this master thesis. An introduction to how technologies can be applied to solve the challenges of the system. A platform for a prototype implementation is described, and I define the five phases of a traffic message. These phases are collection, analysis, validation, distribution and presentation of the event and client data. Several technologies are evaluated, and their advantages and disadvantages are discussed.

Chapter 5 - Prototype Implementation

This chapter presents a prototype implementation developed to test the concepts described in this thesis. Different modules in the system are described, and the system supports automatic detection of traffic problems and manual event reports. Screenshots from the application are included. The first section introduces the modules and their interfaces. The three module sets (the client, the server and the presentation layer) are then described in detail.

Chapter 7 - System Tests, Discussion and Future Work

Chapter 5 contains test results obtained from the implementation, discussion about the technologies and their impact on the implementation, and an introduction to suggested future work in the research

area. The communication between clients and the server, the presentation module and the system design is discussed in detail. The test results are examined, and their practical impact is analyzed.

Chapter 8 - Conclusions

A short summary of the experiences from this project, and an evaluation of the methods used.

Chapter 2

Background

This chapter intends to give a brief introduction to the various fields of research that this master thesis is concerned with. Mobile technology, geospatial standards, content created by users of a service and traffic information are the topics for this section. Mobile devices, available networks and development environments are introduced, together with open standards for the exchange of geospatial information. Weblogs and wikis introduce the concept of including user created content in an application, and is followed by a introduction to traffic information systems, Scalable Vector Graphics and XML-RPC. The chapter ends with some conclusions regarding the background material presented.

2.1 Mobile Computing

Ever since the laptop computer was introduced to the market, mobile devices have become faster, smaller and lighter. Device classes include laptop computers, personal data assistants, mobile phones, handheld game units and in-car computer units. The increase in capabilities for these units can be compared to the evolution of the desktop computer in the 1990s. With the progress towards better devices, we're starting to see a feature-wise convergence towards one common unit. The personal data assistants include phone capabilities, mobile phones are becoming personal data assistants and both are closing the gap to laptop computers. Another introduction to some of the concepts, standards and an overview is located at [9].

2.1.1 Mobile Networks

Mobile networks has progressed from the original NMT450 [10] to the recently launched UMTS (3G) [11] network. The evolution has brought new services to the market, and consumers now have access to wireless networks globally. This section covers GSM, GPRS and UMTS.

Global System for Mobile Communications (GSM)

GSM emerged from a study group formed by the Conference of European Posts and Telegraphs (CEPT) [12], where criteria of interoperability, cost of terminals and speech quality had to be met.

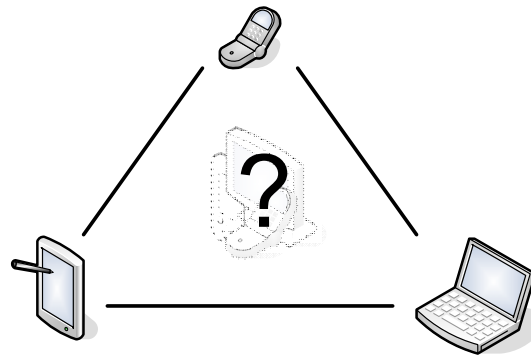


Figure 2.1: The merge of different mobile devices

The first GSM specification was released in 1990, with the first GSM network demonstrated publicly at the ITU's Telecom 91 exhibition [13]. The first commercial GSM networks were available in 1991, but because there were no available handsets, adaption were slow. At the end of 1992 the situation had changed, and handsets were becoming available. GSM was the first mainstream digital wireless network, and encryption is applied to the transmitted data stream. The encryption is based on the A5/1 [14] and A5/2 stream ciphers, and may be replaced by the network operator. The GSM standard also provides services for fax and data transfer, ranging in speeds from 300 bits pr second, and up to 9600 bits pr second. The user is identified separately from the handset, by a personal SIM-card, which is used by the handset to obtain the network and personal details of the user. The GSM network is the only network with almost full coverage of Norway, and the widest deployed standard for mobile networks in the world. Newer GSM networks uses GPRS for data transfer, but in older networks the transmission rate is limited to 14.4 kbps or 9.6 kbps.

General Packet Radio Service (GPRS)

GPRS [15] is a packet based transmission protocol available in existing GSM networks. The spare bandwidth available in an GSM tower is offered to clients for data transmissions, and a client may use one or several channels to obtain better communication rates. The bandwidth is shared among all users at the tower, but GPRS is capable of transmitting data at 171.2 kbps. GPRS allows access to IP-based networks, and most operators have an interface to the Internet. This means that GPRS compliant phones are Internet capable terminals. GPRS is usually priced according to the amount of data transmitted [16] [17], although some operators are now offering flat rate subscriptions.

Universal Mobile Telecommunications System (UMTS)

UMTS [11] is the most recent standard to be deployed in Norway. Featuring higher transmission rates than previous initiatives, UMTS is capable of transferring both video and application data of decent quality. Video conversations are offered as one of the new services, and users are now able to stream live TV [18] [19] on their mobile phone. Video and high resolution images can also be

included in Multimedia Messages (MMS), and sent between handsets. Most available handsets are able to connect to an GSM network as an alternative, because UMTS coverage is only available in urban areas. 2.2 shows the UMTS coverage available in southern Norway at the time of writing.

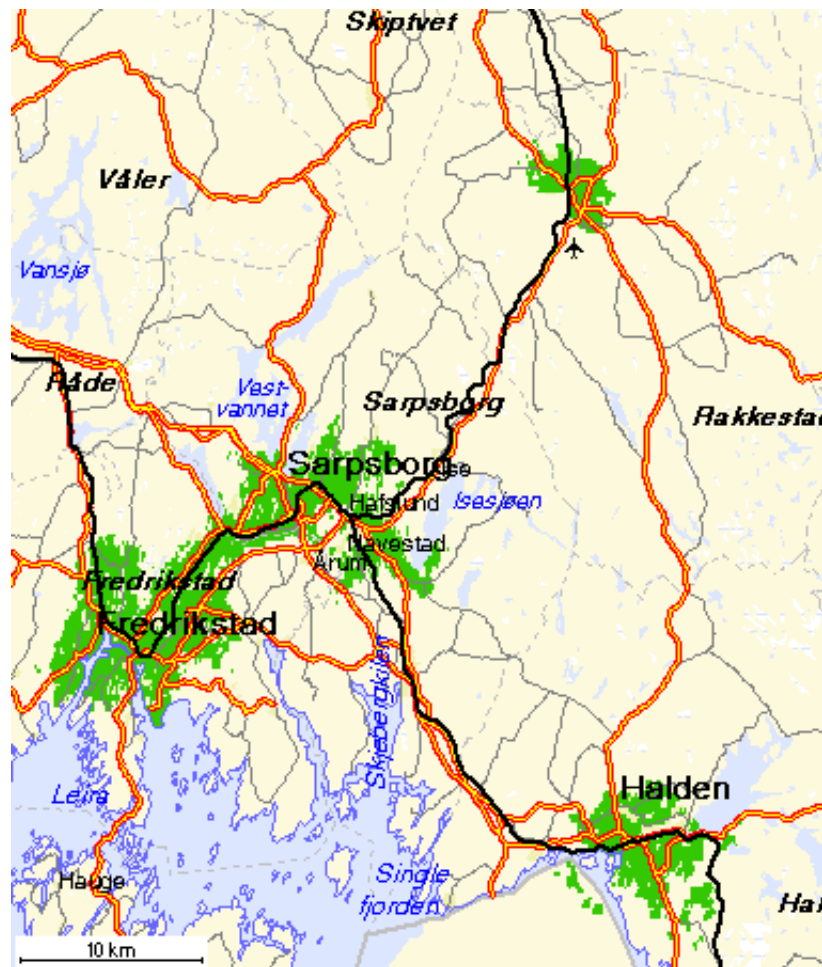


Figure 2.2: UMTS coverage map for southern Østfold, Norway, 22. May 2005

2.1.2 Mobile Application Development

The development environments for mobile devices are closing the gap between desktop computers and mobile units. Today's mobile phones and personal data assistants are capable of performing operations on the same level as desktop computers 3-4 years ago. Because the number of developers has increased, so has the technical solutions available. Programming embedded devices was a challenging task, but the situation now is better. This section contains an introduction to technologies that are significant for developers on mobile platforms.

Platforms

Several mobile technologies has become widely deployed development platforms. These include virtual machines and integrated device operating systems (Symbian, PalmOS, etc.). This section gives a brief introduction to the different platforms.

Java MIDP2 The Java Mobile Information Device Profile 2 (MIDP2) standard is the most deployed API available, and MIDP2 compatible virtual machines are available for all modern mobile platforms. The familiarity to existing Java developers and support between platforms, are the reasons behind the popularity of MIDP2. Because of this, games and entertainment applications are usually developed for this platform. The MIDP2 has a limited set of services available to avoid overloading embedded devices with excessive APIs. There are several optional modules that manufacturers can support on an API basis, including support for capturing and playing multimedia files, access to the file system of a device and communication handling. Because not all MIDP2 compliant devices support all optional features, care has to be taken when developing an application. Figure 2.3 shows some optional modules and their location in the Java hierarchy.

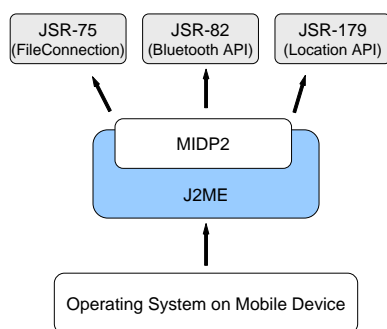


Figure 2.3: Java MIDP2 and some of its optional modules

The Java MIDP2 follows the Java guidelines concerning security, and measures are implemented to stop applications from misbehaving. All applications run in a local security context, and permission to perform certain tasks is requested explicitly from the user (depending on the active security profile). There are commercial code signing programs available for integrating more trust in an application, so that the user can be certain the application will not misbehave.

Symbian OS The Symbian Operating System is an operating system created for embedded devices. These devices range from PDA units (such as Psion's range of hardware) to mobile phones. Symbian is found in high-end mobile phones, where the boundaries between PDAs and mobile phones are being erased. The development language available is C++, and the API provides handling of user interfaces and communication through messaging and sockets. If a developer needs low level access to the hardware of a device, Symbian is probably the best alternative. Symbian provides a virtual machine for running MIDP2 applications, which allows all Symbian based devices to have access to the MIDP2 applications on the market.

See Figure 2.4 for a comparison between the Java MIDP 2 and the security model of applications running in a Symbian based environment.

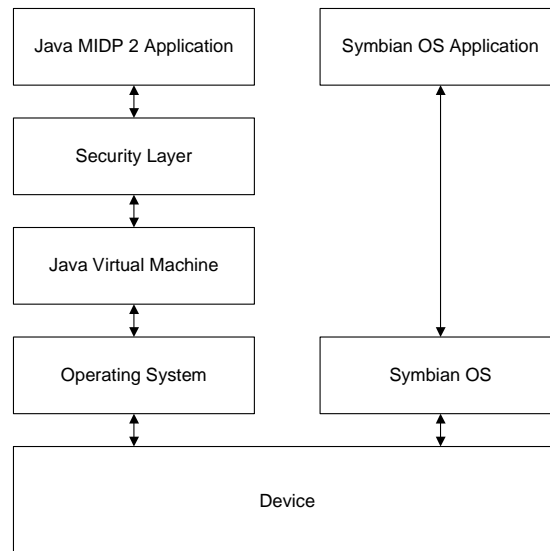


Figure 2.4: A comparison between the Java MIDP2 and the Symbian OS.

Series 60 Series 60 [20] is a Symbian based device profile available in particular on Nokia's phones. The Series 60 API is a standardization of available features, and includes an extended graphical interface, a separate development environment and several APIs that are not available for the standard Symbian OS.

Python for Series 60 Python for Series 60 is a port of the popular Python programming language [21] to the Series 60 platform. The release includes several libraries for handling the mobile context of the platform, and contains extensions not bundled with the original version of Python. Due to the size limitations of mobile devices, the number of included libraries is kept to a minimum. The libraries can however be installed in the Python environment by transferring them to the device by Bluetooth [22] or cable. Python's extendability is still maintained, and custom modules may be developed through Symbian's C++ interface.

PalmOS Originally developed for the PalmPilot PDA, PalmOS is still the operating system on recent Palm handhelds. Their feature set is similar to the set offered by the Symbian OS, although their attention has been towards the PDA market. Development is done in C and C++, but recent versions of PalmOS features a virtual machine for Java MIDP2. An example of how to develop a geospatial application on PalmOS can be found in [23].

Pocket PC Microsoft's operating system for mobile devices is Pocket PC, a platform based on Windows CE - an embedded version of Windows. The Pocket PC platform offers the familiarity of the Windows platform to users and developers, and enables rapid cross platform development and deployment of applications. Programming language support is extensive as Microsoft's own Visual Studio is able to compile its languages to the platform. Pocket PC also supports the Java MIDP2.

Java Personal Edition The Java Personal Edition was the predecessor of the Java MIDP2 and was deployed on high end phones and PDAs. The Personal Edition is a complete implementation of the Java 1.1.8 profile, and has a more relaxed security setting than MIDP2. Because the specification was originally developed for the desktop version of Java, the profile lacks features otherwise associated with mobile devices.

Messaging

Traditional communication on mobile phones has been short text-based messages, low bitrate data channels and voice communication. The messaging subsystem can be described as a way to send small, enclosed chunks of data intended for another mobile device. The two technologies described satisfies different requirements of messaging, and can deliver basic communication services independent of network and application API structure.

Short Message System (SMS) The Short Message System (SMS) was introduced together with the GSM network. A simple communication protocol developed to allow consumers to send small messages to each other, SMS was never intended to reach the current popularity. A message is limited to 160 characters, and can be used for sending remote commands to a mobile device in addition to communication between individuals. Numbers from the Norwegian network operator NetCom [17], indicates that the the amount of messages sent in the last four years increased six fold.

Multimedia Messaging Service (MMS) With the advent of multi-functional mobile phones, the need for an advanced messaging system became apparent. The network operators created The Multimedia Messaging System to fill this need. Connections are established through the Wireless Access Protocol (WAP) [24] and uses the network's transport protocol (i.e. GPRS, EDGE, etc.) for data transmission. The binary transfer allows for remote storage and retrieval of media independent of type. The exchange of MMS messages between handsets is done through a central server provided by the network operator. The receiving phone is then requested to connect to the MMS server by a special SMS message. MMS message can be annotated [25] with the location of the user.

File Access

The mandated security limitations of the MIDP2 virtual machine is an obstacle for developing integrated applications for mobile devices. After the release of JSR-75 [26] there is a proposed standard

for handling file access through an optional MIDP2 API [27]. Recent high-end Nokia phones support this feature, and the technology makes the development of more integrated applications available on the embedded profile of Java. The Symbian operating system also supports low level file access, and the file system can be manipulated as on desktop computers.

Multimedia Access

With the introduction of multimedia phones came the need for programmatic access to the capabilities. An application should have access to the camera of a device, the speaker system, the microphone and other dedicated hardware. This allows for the development of applications that utilize the full potential of a phone, including the graphical and audible features. The MIDP2 profile supports the access to multimedia capabilities through an optional extension named MMAPI (JSR-135) [28]. The Symbian operating system has its own multi-threaded library, the MultiMedia Framework (MMF) [29]. This framework allows for the creation of codecs: small library functions that encode and decode certain media formats. This is how most media frameworks are structured, and allows developers to add support for new types of media to existing applications. Symbian supports the manipulation and retrieval of data from the internal camera through the Ecam interface.

Personal Networking

Another issue brought forward by this generation of mobile devices, is the need for a personal network. Connections between the local device and external units have traditionally been by cable, but with the increase in available external units, a wireless exchange protocol was needed. Personal networks are wireless, have short range and are based on low power broadcasts. The industry standard for personal networks is Bluetooth [22], a protocol for communication between auxiliary devices. The Java MIDP2 supports access to the Bluetooth capabilities of a device through the BTAPI (JSR-82) [30] interface, an optional extension. This allows developers to use the personal network capabilities of the phone to communicate with external units, such as a GPS device or a wireless handset [31].

2.2 Open Geospatial Specifications and Services

The introduction of Geographical Information Systems (GIS) in the world of cartography increased the ability to use and maintain geospatial data. GIS also enabled non-cartographers to use the geographical data in their projects, and developed a better understanding of the relationship between information and location. During the introduction of Web services in the 1990s, it became apparent that proprietary interfaces was hindering development. There was a need for common set of protocol specifications, and the Open Geospatial Consortium (OGC) [32] specified a set of protocols to solve this problem. The protocols define sets of functions that can retrieve and manipulate geographical data. This initiative is based on the initial development of an XML based grammar for the storage of geographical data, the Geography Markup Language (GML).

2.2.1 Geography Markup Language (GML)

The GML format is an XML [33] based markup language for describing geographic features. The language has been evolving since its first release as a public request for comments (RFC) on December 16, 1999. The latest release, GML3.0 [34], supports topological structures, embedded measurement data and extendibility through XML. By using existing metadata frameworks, the information can be augmented with additional information in a well-known format. One such XML grammar is the Dublin Core [35] standard, which is further described in Section 2.4.4.

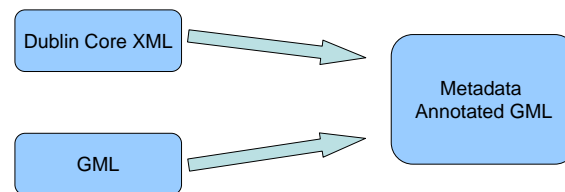


Figure 2.5: GML extended with metadata information in the Dublin Core grammar

GML is an open specification to facilitate the storage and exchange of geographical information. The deployment of GML has gained an considerable amount of momentum since its release [36]. The Semantic Web [37] has become one of the key concepts for describing information on the World Wide Web, and GML allows geographical information to be used and exchanged in an open format. Most applications use GML as their language for describing geographic features, and use GML to extend existing information. By implementing GML as the storage grammar for geographic features, interoperability between GML compliant applications is preserved. In addition, the information is still human readable as a consequence of the verbosity of XML.

The GML3.0 standard tries to encapsulate almost all geographical information, and is not limited to primitive location coordinates. The range of observations representable in GML3.0 documents are extensive, and may include aspects as direction (See Section 7.11 in [34]) and observatory information (Section 7.12 in [34]). Observations are in particular valid concerning the observatory nature of this thesis, and can include a location, a time instant or a time interval, in addition to the result of the observation. The GML3.0 specification doesn't limit the result from an observation, so the user may embed or refer to different types of data. An example of an observation can be the act of taking a photograph, which has a direction, a motive (the observation), a location (where the camera is) and a result (the picture).

Although observations can be used to describe an image, they can also include other events. Another example would be traffic related information, where there may be both instant (an accident) or interval based (roadwork) information. The GML3.0 standard can represent both these event types. Different measurement units are also represented in the standard, which allows for the use of native units depending on where the application is implemented. Other implementors can then be aware of the units of measurements, and can adjust the presentation to the user. This is further described in Section 7.10 of [34].

2.2.2 Web Map Service (WMS)

The most common use of maps is as raster images. An area of the world is presented to a user as an image, either for annotation or for presentation. The Web Map Service (WMS) provides a standardization of the request parameters and error handling in such a system. The WMS standard is an open and freely available standard from the Open Geospatial Consortium (OGC), and the specification is ISO-certified [38]. WMS is considered to be the de facto standard for interchangeable raster map images on the web. Several sites (such as the Arealis Project [39]) use WMS to deliver their content in an user accessible manner. By enabling users and researchers to extend their information with real time maps suited to each individual need, the WMS technology has proven to be usable and practical to implement [40].

The WMS standard is based on the same interface as other OGC standards, where the parameters are presented either as a HTTP GET or a HTTP POST request to the web service. The parameters are almost identical between services, and the important parameters to the Web Map Service are the `Layers`, `Styles` and `BBox` parameters. These describe the information the resulting raster image should contain, in particular which thematic layers (See Figure 2.6 and Figure 2.7) should be present, how they should be formatted and for which geographical area they should be retrieved. In addition the server supports parameters for the size and format of the resulting image. These are the technical details that a user or a developer need to manipulate to use a Web Map Service.

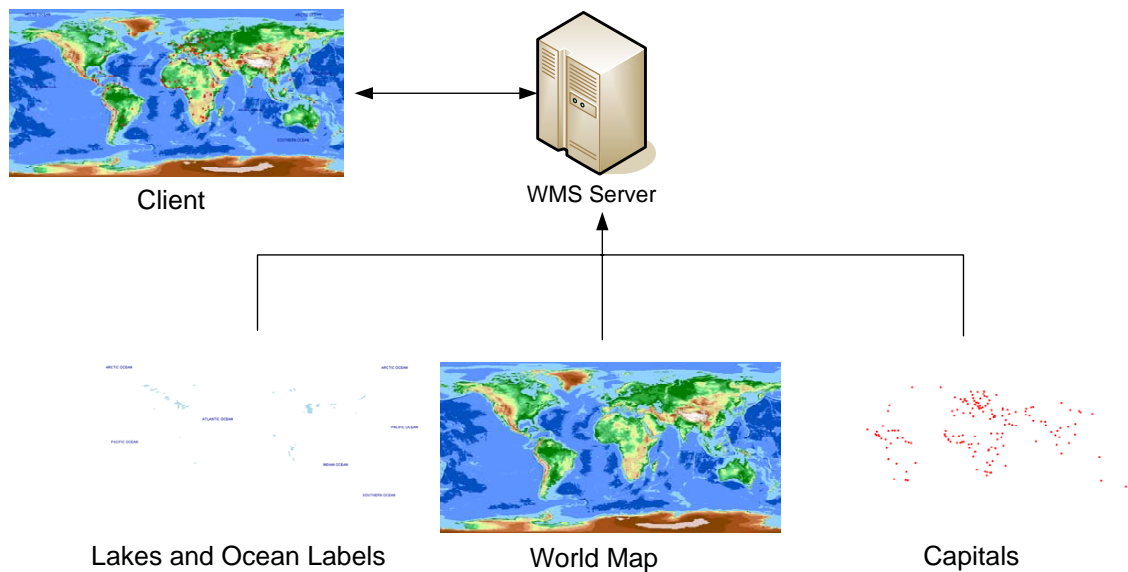


Figure 2.6: WMS server structure

The Web Map Service also supports the aggregation (See Figure 2.8) of other Web Map Services, where the server is responsible for acting as a proxy between the user and other services.

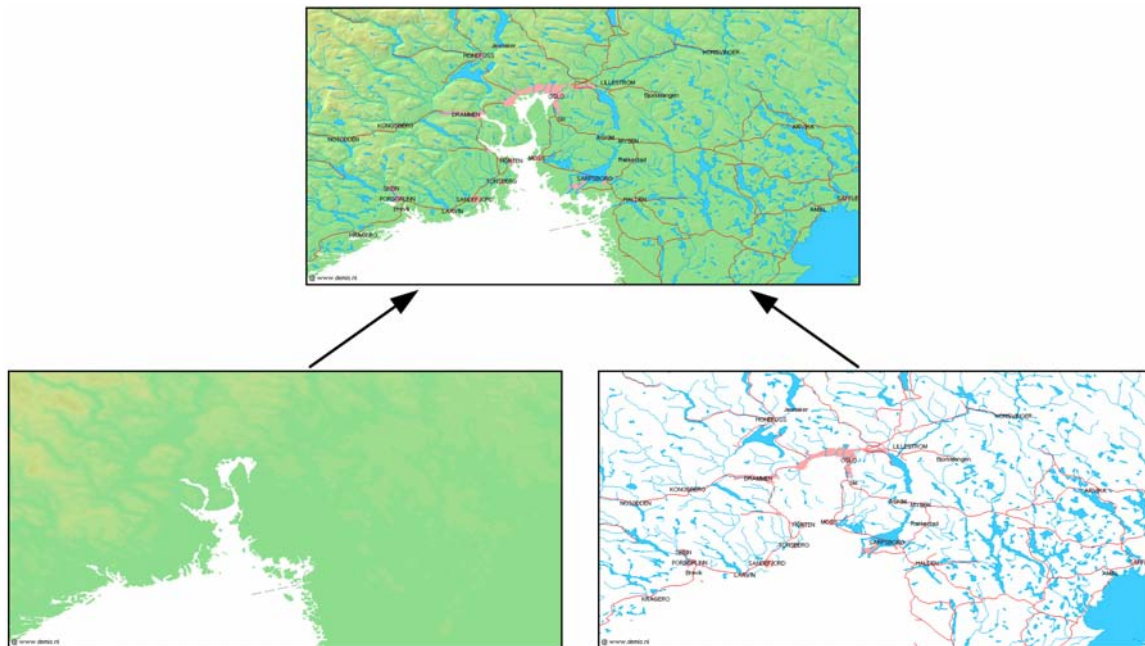


Figure 2.7: WMS layer merging

This allows the user to only be familiar with one local WMS. Because of the interoperability requirements of the protocol, and because only available information is present in each layer, several services may be merged to form one uniform response. A road navigation system may be concerned with roads, speed limits, toll booths and driving directions. Some of this information is irrelevant a researcher creating a map showing pollution in swamp areas. The map can be adjusted to the requirements of that particular user, by removing and adding layers of information. The data sources are still located at the entities providing the data, and the local WMS server retrieves the external information transparently.

Programmatic interoperability is provided through the same method as other OGC services, where the service responds to a `GetCapabilities` request. This request returns an XML document describing the data the server provides access to, which formats it can present its information in and which coordinate systems are available. This enables the development of autonomous clients which may adjust their interface depending on the available layers and styles at a WMS. Interoperability is one of the main reasons for the Web Map Service, and it's definitely the cause of its wide spread usage. In addition to the possibility of retrieving raster images from a Web Map Service, the WMS specification defines an optional query for feature information at a particular image coordinate (See Figure 2.9). The information returned is application specific and free in both content and structure. This feature may be used to develop dynamic information systems, where the WMS provides an interface to georeferenced data.

There are several WMS server applications available, both as commercial solutions and open source projects. Demis [41] is one of the commercial vendors, and offers a WMS server together

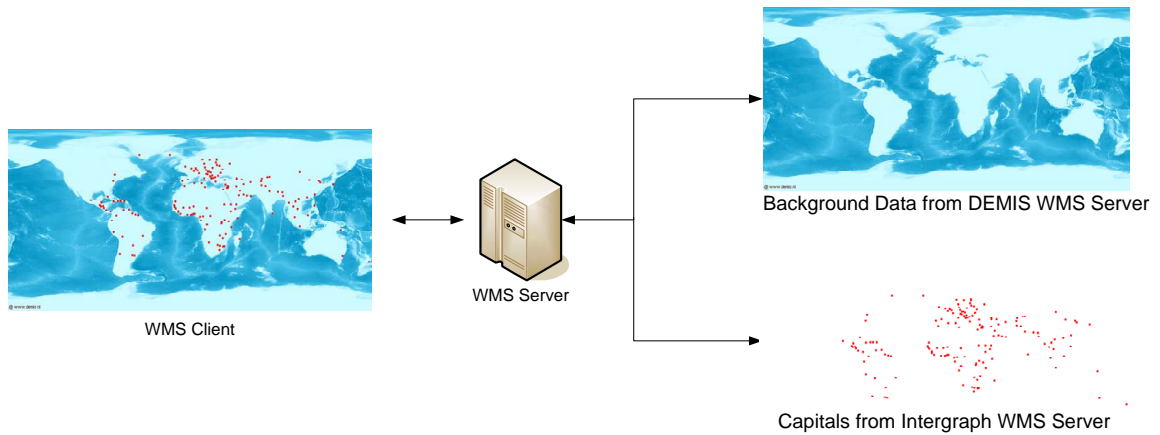


Figure 2.8: WMS server aggregation



Figure 2.9: WMS GetFeatureInfo functionality

with applications for processing data available from a WMS. ESRI [42], sells and deploys their ArcIMS [43] solution which is able to provide content according to WMS specifications. Available open source projects include GeoServer [44] and MapServer [45].

2.2.3 GPS Exchange Format (GPX)

Described as a lightweight XML based format for the storage and exchange of GPS data (waypoints, routes and tracks), the GPS Exchange Format (GPX) [46] has become a widely used standard. The intention is to present a simple and easy to use format that applications may use to store GPS data. The format is not intended for the storage of geographic features, but movement paths, single observations and similar information. The GML3.0 format does support these observations, but the extent of the GML3.0 standard makes it difficult to implement in small applications.

The GPX 1.1 specification only supports the storage of latitude, longitude and altitude in the description of waypoints, although the GPX 1.0 specification also supported two elements for speed



Figure 2.10: GPX used as a transport format

and course (heading). This change was not intended ¹, and the elements describing speed and course should return in version 1.2 or as additional application schemas. For further information about the format, please refer to the GPX Web site [46]. Several applications have been developed for the GPX format, mostly by the geocaching [47] community. Examples of such applications are Spinner [48], GPXView [49] and CacheMate [50]. Several sources are also available for the publication of waypoints and tracks in the GPX format, such as magnalox [51] and TrailRegistry [52]. These applications are lightweight programs that only provide a subset of geometric operations.

An alternative to the GPX format is the POIX [53] XML grammar, developed and specified as a World Wide Web Consortium (W3C) [54] standard. The grammar has a broader description of collected points and is tailored towards complete observations. Concerns have been raised about the format, and implementations are missing. The initial version from 1999 is still the current one, and the format can be considered dead.

2.2.4 The NMEA GPS Protocol

The currently used protocol by most GPS units is a protocol described by the National Marine Electronic Association (NMEA) [55] (NMEA-0183). This is a simple text based format where each line contains a comma separated list of values. The first field identifies the type of information contained within the line, while the other fields are the actual data. Appendix A.1.1 contains examples of several different NMEA strings. Strings that start with \$GPRMC indicates that the data is of the recommended minimum type. This pattern features properties such as the position, velocity and heading of the receiver. Other strings identifies the number of satellites in the view, the information delivered by these satellites and the dilution of precision.

2.3 Location Based Services

Location Based Services is the area where the technologies of mobile devices (Section 2.1) and geospatial standards (Section 2.2) meet. During the last five years the idea of location based services was introduced to the mobile industry and the average consumer. In the dotcom-era of the late nineties only mentioning the words location based service was enough to receive venture capital. Those days have passed, and many consumers are left with the impression that location based services never reached market.

¹See <http://groups.yahoo.com/group/gpsxml/message/734>

2.3.1 What is a Location Based Service?

A location based service is a service that considers the physical position of the request when determining the result of the service. An example of this is the "Find the nearest .."-queries, where the user requests information that is relative to his current position. A result from a query can indicate all Chinese restaurants in a particular area, or the closest 24-hour open chemists. While this may seem like a unique service, it is the same problem which applies to other geospatial queries such as "Find chemists in Los Angeles". It is important to remember that location based services are the same as traditional geospatial services, but they have a precise location affixed to the query. Today's location based services have been considered one way only, which means that the user is only acting as consumers in the implementation. To determine the current position of the phone, several different technologies can be applied, each with its own advantages and disadvantages.

2.3.2 Introduction to Mobile Positional Methods

This section should not be considered a general guide to the area of how positioning methods work, but is only intended to give a introduction to the different forms of positioning. It will not give a technical introduction to the field of positioning technology.

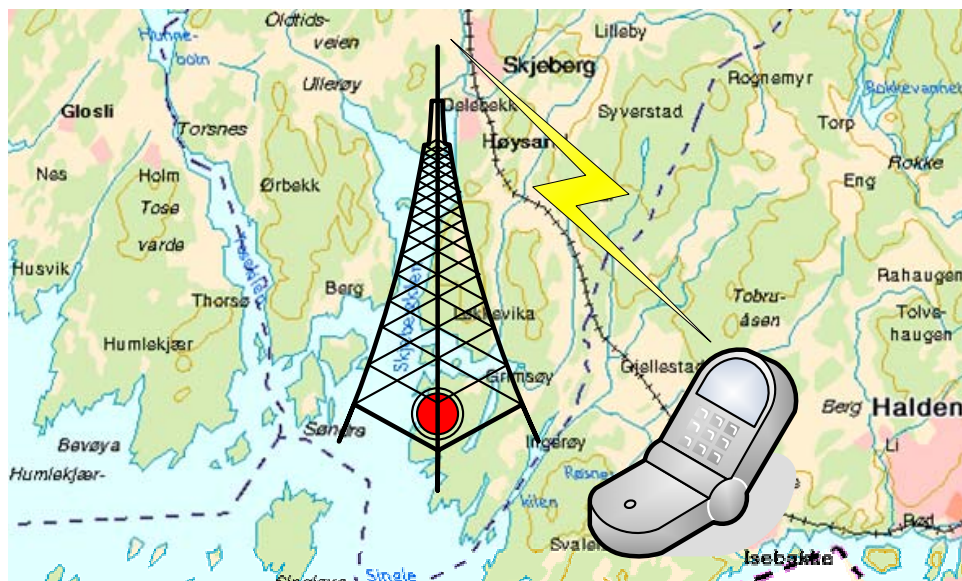


Figure 2.11: Determining position by cell tower location

The most common method of mobile phone positioning, is to look at which cell tower the phone is connected to. This information is either available from the phone, or from the network operator (often associated with a fee). While the phone has no means of knowing the location of the cell tower, a central registry can be maintained which can resolve unique cell ids to latitude and longitude coordinates. Databases of this type is not available to the public from the network operators, but

several grass root projects [56] attempt to build such databases from data gathered by volunteers. This would allow for cheap and efficient applications, without the need for a third party service provider. The advantage of this method is that it is available as long as the phone has a connection to the network. The problem is that if the phone loses reception, the spatial dimension disappears. The accuracy of the position also leaves a lot to be desired. Depending on the environment the cell towers are located in, the inaccuracy may be up to 10-15km in rural areas. This allows developers to locate a phone within an area, but it is not suitable for narrow queries.

Measures can be implemented to increase the accuracy of the location determined by the cell tower. The most common method is to use the timing advance (TA) [57] information delivered by the phones and the network. In combination with determining the direction of origin of the signal relatively to the cell phone tower, one is able to locate the phone more accurately than by cell tower id alone. This requires custom processing of the signal by the network operator, and may require custom equipment installed in the cell phone towers. Further information about future technologies, including E-OTD and A-GPS, can be found in [57], [58] and [59].



Figure 2.12: Mobile smartphones and Bluetooth equipment

The easiest way to determine the position of a mobile phone is to let the phone report it. The phone can be equipped with a Global Positioning System (GPS) [60] receiver, and can be interfaced with the phone by Bluetooth or a serial connection. This enables applications on the phone to be

aware of their current position. Many future phones will be equipped with GPS receivers because of legislation concerning the information reported to emergency services if an accident occurs [61]. The accuracy of the position retrieved is reliable down to 10-50 meters (even better with a differential GPS [62]) and doesn't require a network connection to work. It does require a clear view of the sky and is prone to the problem described as "Urban Canyons" [63].

2.3.3 OpenGIS Location Services (OpenLS)

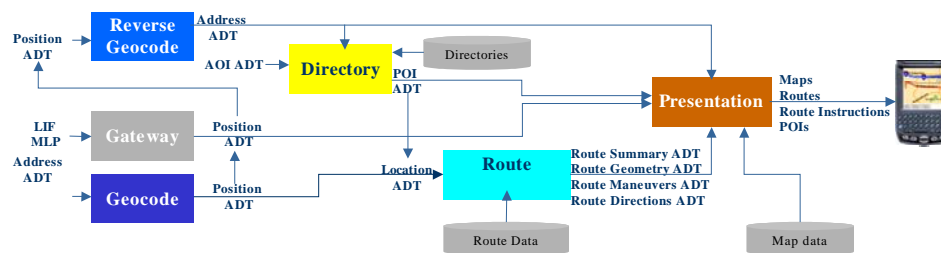


Figure 2.13: OpenLS information model (from [1])

The Open Geospatial Consortium (OGC) has released a specification detailing the Core Services that should be offered by a Location Based Service, and how the queries and their responses should be formatted. The interface is named OpenLS, and is a complete set of XML schemas describing the queries, the responses and error handling of the system. The OpenLS Core Services document [1] describes the five key services as the Directory Service, the Gateway Service, the Location Utility Service (geocode/reverse geocode), the Presentation Service and the Route Service. These services work together to create a functional location based service.

Directory Service

The Directory Service provides answers to queries like "find the nearest" or "find a", where a directory keeps track of locations that users may take interest in. The service has two distinct operations, the pinpoint service and the proximity service. The latter provides answers to "find the nearest"-queries, while the former provides a location for a type of service or a named entity. Both can result in zero or more coordinates being returned, depending on the location and the type of the query.

Gateway Service

The Gateway Service maintains the connection between the location service and the mobile network. The position of the user is determined by The Gateway Service, and is then reported to other subsystems.

Location Utility Service

The Location Utility Service is responsible for doing geocode and reverse geocode lookups. Examples of such queries are "What is the coordinates for Times Square?" and "What road is close to 57.38,11.12?". The former is a geocode service, while the latter is a reverse geocode service. Another related service is a gazetteer, where users may look up locations and have them resolved to a set of geographical coordinates. A proper geocode service is a vital part to integrate natural language queries in a system, and allow human references to objects.

Presentation Service

The Presentation Service is responsible for presenting the result to the user. This is where the information collected and calculated by the different modules end up. This subsystem is then responsible for presenting that data according to the user's visual preferences, such as a route description, a graphical map, images of visual landmarks, etc. This is then presented to the user as the result of the original query.

Route Service

The Route Service calculates routes between different geographical points. These may be limited by particular properties, such as type of transportation, hour of day and location of the user. The Route Service provides these calculations as an integrated part of the system, but may use an external route planning service. The result is sent to the presentation service.

2.4 User Generated Content

The intention of this segment is to give a brief introduction to the different mainstream concepts that use user generated content. Several community projects build on the knowledge of their users, and enables individuals to reach an audience.

2.4.1 Blogs

At the end of the 1990s, a new concept of semi-interactivity started to spread across the net. The "weblog" or "blog" for short, contains references to other Web sites and short comments [64]. Later this crossed into the area of the traditional journal, where a user would post his thoughts. Blogs throw references back and forth between each other, creating small ad-hoc networks (See Figure 2.14 and [65]) that implies a connectivity between individuals. These communities are examples of how people within field of interest create small, non-formal groups which exchange and discuss information.

The blog concept has evolved into several new initiatives such as citizen journalism. The lack of control by one media corporation has made it possible for anyone to reach out with their views and commentary. A large amount of people read blogs every day, and several of these people also maintain their own blogs. This collection of people features a diverse set of knowledge, skills and abilities. By annotating the information they receive from each other, a large collective community

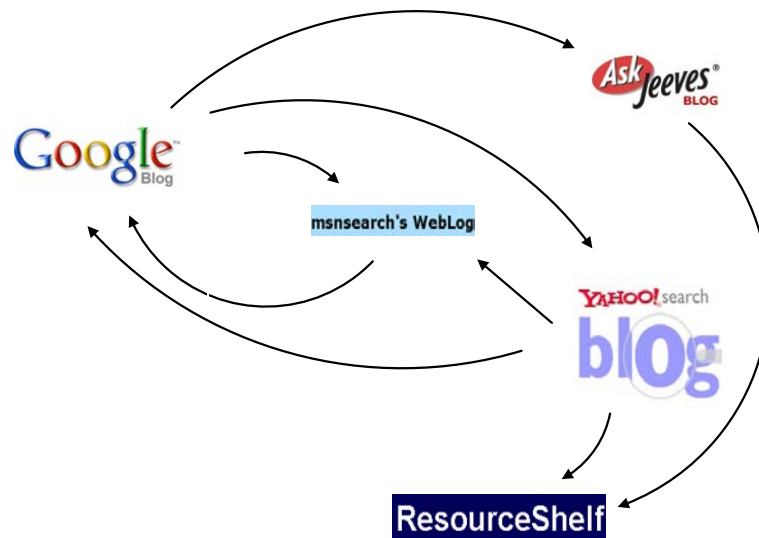


Figure 2.14: The inherent network structure between blogs

may be able to provide more information and a more thorough focus on situations than traditional media [66]. During the last year blogs have become mainstream, and most U.S. media corporations now feature some blog-based information.

2.4.2 Wikis

While blogs are usually maintained by an individual, the wiki concept invites everyone to be involved. A wiki is a constantly evolving Web site, where the content is defined by the users. If a user has something to contribute on a subject, he updates the page to reflect his view. There is no communication between the user and a publisher or an editor, and the change is automatically committed to the database. This enables the site to use the experience and knowledge of their visiting users, and not only of the initial publisher of the material.

The worlds largest, independent encyclopedia is built by this method, and is named Wikipedia [67]. The only possibility for Wikipedia to reach the size it has, is by building on the knowledge of its users. If someone has knowledge about a subject, he or she is able to submit that information. Later someone else reads the information and extends or corrects errors in the previous submission.

While there seem to be serious implications about allowing anyone to edit and change the content of a web page, the current system implemented at Wikipedia shows that the system can work. There are measures in place to detect fraudulent behavior, such as spamming, changing small details of articles and submitting erroneous information. There are active editors monitoring the repository, and able to revert changes. By having a version control system, an article can be reversed to the previous version.

The non-authoritative nature of blogs and wikis requires a more sceptical view than one would

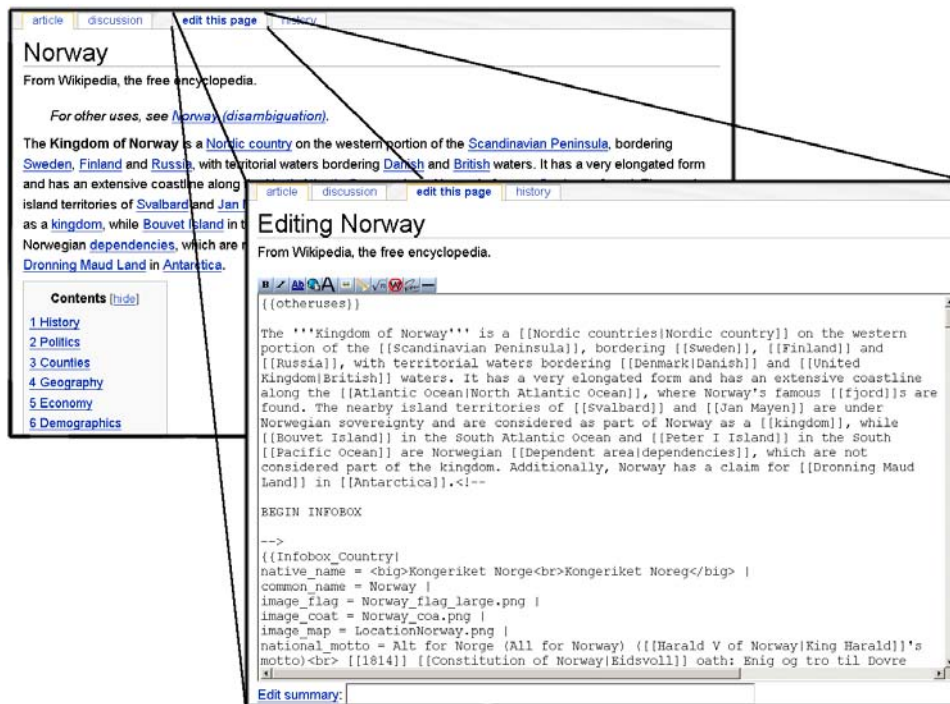


Figure 2.15: Editing a page on Wikipedia

expect for traditional media, but these boundaries have been erased during the last years after revelations about massive factual errors by publishers [68]. The requirement of active and conscious editors are certain, and a critical approach to new content should be maintained.

2.4.3 Citizen Journalism

After the introduction of the blog in the 1990s, a new term was defined: citizen journalism. The main inspiration is Dan Gillmor's book "We The Media" [69], which has inspired several initiatives (such as Wikinews [70]). The concept of citizen journalism is to allow people to act as news reporters. They capture images and video, write texts and communicate with other people who are interested in the same area. The information flow depends on which reporters you follow and read content by. Because no person or organization can monitor the whole world, people write about what they feel are important to them. A similar concept is described in [65].

One norwegian newspaper [71] has experimented with allowing non-reporters to send multimedia images depicting news events as happen (See Figure 2.16).

[Kjenner du noen - eller er du selv - rammet av legionella-smitten i Østfold-byene?](#)
TIPS VG NETT: MMS/SMS: 2200 ✉ E-post: 2200@vg.no
☎ TLF: 22 00 00 00
[Les hele saken](#)

Figure 2.16: Screenshot from Verdens Gang (VG)

2.4.4 Metadata

While data previously went through a validation process, the change to more user provided content created the need for standardized metadata storage and retrieval. Several methods and standards have been introduced, and a large amount of research has been performed within the field of metadata. The most widespread metadata standard is the Dewey Decimal Classification System (DDC) [72] which is used to classify books in modern libraries. The system was developed in the 1870s and provides a dynamic structure for the classification and organization of library collections. The system is divided into sections, such as section 720 which contains books relating to architecture. This section is divided into subsections, such as 720.22. The DDC system can be used to retrieve the exact classification of the book, and books that deal with the same topic are organized in the same class.

One problem is that the DDC is only concerned with the classification of the item, and do not include other metadata. Examples of metadata elements are the name of the author, the publisher, publishing year and other properties depending on the application or the storage requirement. The W3C [54] has launched an initiative called "The Semantic Web" [37], an attempt to extend the information on the World Wide Web with descriptive metadata.

“The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. It is a collaborative effort led by W3C with participation from a large number of researchers and industrial partners.” – World Wide Web Consortium, The Semantic Web Web Page

“The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation.”

– Tim Berners-Lee, James Hendler, Ora Lassila, The Semantic Web, Scientific American, May 2001

The Semantic Web initiative includes several standards, such as the Resource Description Framework (RDF) [73] and the OWL Web Ontology Language (OWL) [74]. Both are used to describe content, and to put the document into a context which can be used programmatically and manually. The final approval of the two standards was announced in a press release at the 10th of February 2004.

The Dublin Core Metadata Initiative (DCMI) [35] is another XML based metadata standard used in applications. The Dublin Core initiative attempts to construct a general, portable format

for extending other data types with a common set of metadata. Many XML based applications has adopted the Dublin Core standard as the preferred set of metadata, and Dublin Core has gained widespread usage. The properties include items such as author, creation date, publishers, contributors, languages and source. Although content include metadata, the metadata is seldomly used in applications. There are two exceptions from this practice, where two different metadata standards have gained widespread consumer adoption and is contained in almost any file of the two formats. The first is the ID3 [75] tag located in MP3-files, and the second is the ExIF [76] header contained in most image formats from digital cameras.

The ID3 tag contains a description of the audio file, and include items such as the artist, the album the track is from, track number, comments and the genre of the music. Although the first ID3 standard was limited and not extensible, it was present in almost any MP3 based file. This lead to a new standard, the ID3v2, which features a richer set of metadata. ID3v2 also provides facilities for capturing the essence of the encoding process and the recording session, in addition to other metadata regarding the content.

The ExIF information is a set of metadata values embedded in several image formats. The goal of the ExIF header is to provide a set of both technical and social data relating to the image, such as the camera setting, the exposure used and items like position and encoding methods. The ExIF header also supports social properties such as the name of the photographer and a description of the image.

Research in the field of metadata has been extensive, because it affects many separate areas of interest. Metadata is present in database structures, media indexing, government registries and within information theory research. Previously metadata was provided or retrieved by a central authority, such as a librarian or a cataloguer. With the advent of The Semantic Web, the extraction and automatic generation of metadata has become an active field of research. In particular, projects relating to the use of metadata in a mobile context ([77], [78]) has attracted research. Interesting metadata in this context is the location and image properties. The automatic discovery of objects and persons in close vicinity also provides a set of metadata [79]. The idea is to free the user from the responsibility of providing the metadata, and rely on the automatic detection and retrieval of information ([80], [81], [82]). These systems are tailored to the application context they are used in, although sets of metadata structures and properties can be shared between problem areas.

2.5 Traffic Information and Car Navigational Systems

Traffic information has been the main mobile content of radio stations through the last twenty years. Information from the authorities and the listeners are distributed through radio shows to inform drivers in problematic areas. Using the characteristic broadcast structure of radio transmissions and the introduction of the Radio Data System (RDS) [83], information can be distributed to the public in different forms and limited by location. Many cars now come with a car navigation systems pre-installed, and consumers install navigation systems in their existing cars. Some systems also supports digital distribution of real-time traffic information, and is discussed in this section.

2.5.1 Extracts from Car Navigation Systems Research

The area of car navigation system research has received much attention, because of the commercial properties and the popularity among consumers. Traffic never stops, and the main choice for transportation is by road. Blaupunkt [84] refers to their car navigation system from 1989 as the first product on the European market. Because Blaupunkt also introduced the first RDS car radio in 1988 and showed the first prototype of a route guidance system in 1982, they have an extensive list of research and prototype implementations for the market. Their latest products include the first online car navigation system from 2002 and their milestone of one million installed car navigation systems in 2001. Since Blaupunkt introduced the first European car navigation system in 1989, the evolution of technology has introduced hi-speed wireless networks and more computational power.

Important properties about car navigation systems is that the user is no longer in a controlled environment. The user's focus has to be on the road, and a car navigation system should not interfere with the operation of the vehicle. Displays have to be simple and easily understandable, and information should be presented both visually and audible [85]. Certain user groups have their personal preferences about how a user interface should be designed, and some commercial operators need to report audible messages back to the operations center. Another important issue mentioned in [85], is the risk of information overflow. In this situation the user is no longer able to take advantage of the information, and the navigation system has become a liability.

After the driver has received and understood a message, the driver should be able to use that message for something useful. If we make the assumption that all messages are of interest to the user, [86] shows that independent of the drivers age, hazard information is remembered in 90 - 93 % of the tests. Junction information was forgotten, or interpreted wrong, in 39 - 57% of the cases. The ability to remember a message for 1-3 minutes was as expected affected by the age of the message (23% errors after one minute, 33% after three minutes). The system used for performing these tests where a visual only system. A similar test was also performed with an audible system [87], and the results shows that audible messages also affect the performance of drivers. The result shows that the quality of the audio was the main reason for performance deviance.

This shows that users have a good chance of remembering information that is considered important, but this assumes that the information broadcasted is correct. [8] shows that drivers are more likely to follow the recommendations of the car navigation system in unknown locations, and that the accuracy of the information affects the drivers trust in the system. Information at a 71% quality level is still considered acceptable and useful, while information rated at a 43% quality level affects driver performance, and the drivers trust in the system. The results indicate that information accuracy below 71% is not recommended, but further research will have to be performed to investigate the gap between 43% and 71%. The recovery of trust in the system was also affected by the driver's familiarity of the area, and negative system advice regarding familiar areas had a greater impact than in non-familiar settings.

Research has also been performed in the field of simulating driver behavior, and the ability of car navigation systems to determine which information the driver needs. Research [88] has shown that there may be a need for a new paradigm for modeling such agent networks and the decision-making strategies of humans in traffic situations. Research that borders on the same areas of mobile traffic information as this thesis has also been performed [89].

Another research project concerned with traffic information has developed a system named SOTIS [3], a self-consistent network between cars on a road segment. SOTIS is an attempt to create a network that is independent of a central authority for the distribution of information to clients. The problems described in the paper is the delay in information propagation, the need for a large number of participating vehicles and no external communication.

2.5.2 Current Industry Standards

The current industry standard for the distribution of traffic information through radio is the traffic announcement subsystem of RDS, called RDS-TA. RDS-TA is a flag that is enabled or disabled in the RDS stream, and tells the receiver whether the information is traffic related. Receivers may highlight RDS-TA content by pausing a CD, switching to the radio channel or increasing the volume. This is a suitable distribution method for the listeners, but makes it hard to use the information programmatically. Digital Audio Broadcasting (DAB) [90] was recently introduced on the market, and supports digital distribution of data. The European Broadcasting Union (EBU) [91] uses a format developed by the internal Transport Protocol Experts Group (TPEG) [92]. The format is named TPEG, and is distributed in binary form and as an XML grammar. TPEG is independent of the distribution method, and can be used in different contexts.

TPEG is a recently developed standard, and is still gaining acceptance with the members of the European Broadcasting Union. An implementation by the Norwegian broadcaster NRK [18] is shown in Section 2.5.3, and uses TPEG as the data grammar. Another live example is available from the Travel News section at the British Broadcasting Corporation's (BBC) Web site [93]. The TPEG Web site currently shows that they are still authoring grammars for the distribution of parking, congestion and travel-time information. Environmental data and weather information is also planned extensions, but no time-line is available. Older systems for the exchange of traffic information is DATEX and the US ITS architecture. DATEX has become well-known for its problems with the interpretation of the standard [94]. In Japan research is focused on the Road Web Markup Language [95], an XML based grammar for traffic information. The American Society of Automotive Engineers' (SAE) [96] technical committee has described the "Advanced Traveler Information Systems Committee Project":

“The purpose of this work plan is to develop a minimum set of medium-independent messages and data elements needed by potential information service providers (ISP's) to deploy ATIS services, and provide the basis for future interoperability of ATIS devices.”

2.5.3 Current Industry Practice

This section will contain some information about the current industry practice in Norway, and the implementations that are in use.

There are two large providers of traffic information to the public in the Norwegian market segment. NRK [18] is the national broadcaster, and features five national channels, two for tv and three for radio. They have long traditions within traffic information, and they were one of the early implementors of the RDS standard. They are currently the only provider of RDS-TA information in

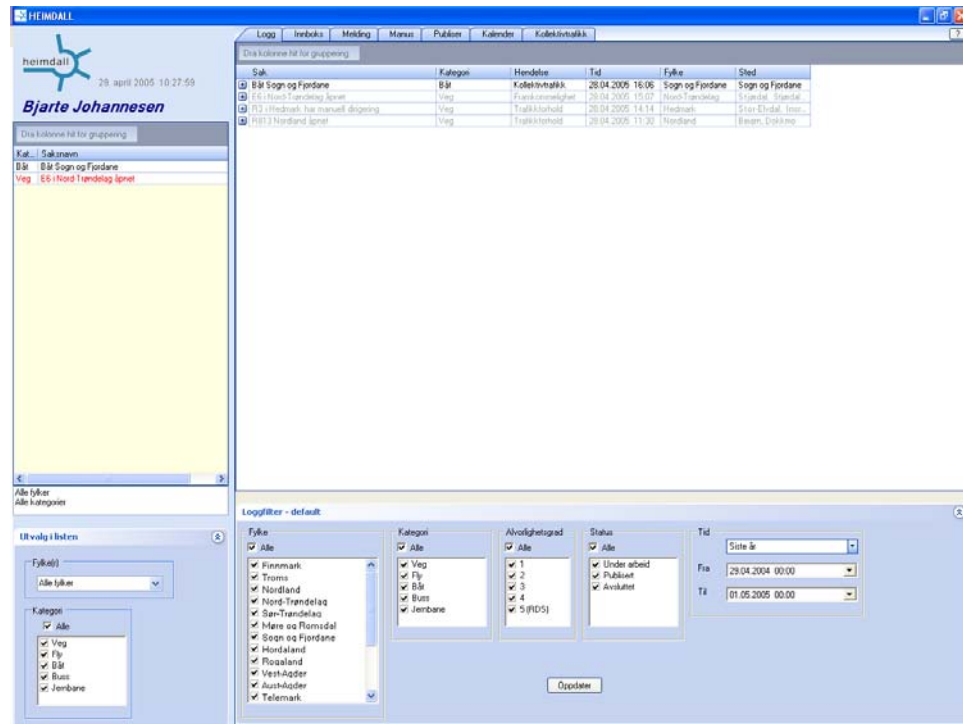


Figure 2.17: Screenshot from the Heimdall application

Norway. Only major events that affect a large number of people are broadcasted with the RDS-TA flag set, so that people understand that the message is important if they receive a RDS-TA broadcast. More information about NRK is available later in this section. The other large provider of traffic information in Norway is the commercial radio station P4 [97]. They have traffic information as one of their specialities, but does not offer RDS-TA. The third, smaller provider of traffic information is Kanal 24 [98]. They are still a recent addition to the radio segment in Norway, but do also feature national coverage. They were unable to comment on their current equipment and plans for the future.

Both NRK and P4 stated that most of their traffic information comes from the authorities, such as the police and the Norwegian Public Roads Administration. The latter provides information regarding closed roads, planned road work and similar events. The police provides spontaneous information, such as accidents, large congestions and situations where public safety is at risk. P4 and Kanal 24 has traffic helicopters active during the week, and monitor the traffic around the largest cities in the south-eastern part of Norway. The information is only provided as audio segments in their regular programming. P4 stated that between 20 and 40 broadcast messages during the day originates from their listeners. NRK relies on their information partners, and reports originating from listeners makes up 2-3% of the total amount of messages broadcast.

A Norwegian initiative named the National Road Database [99] attempts to gather information about the Norwegian roads in one database. The first version went public during the spring of 2005,

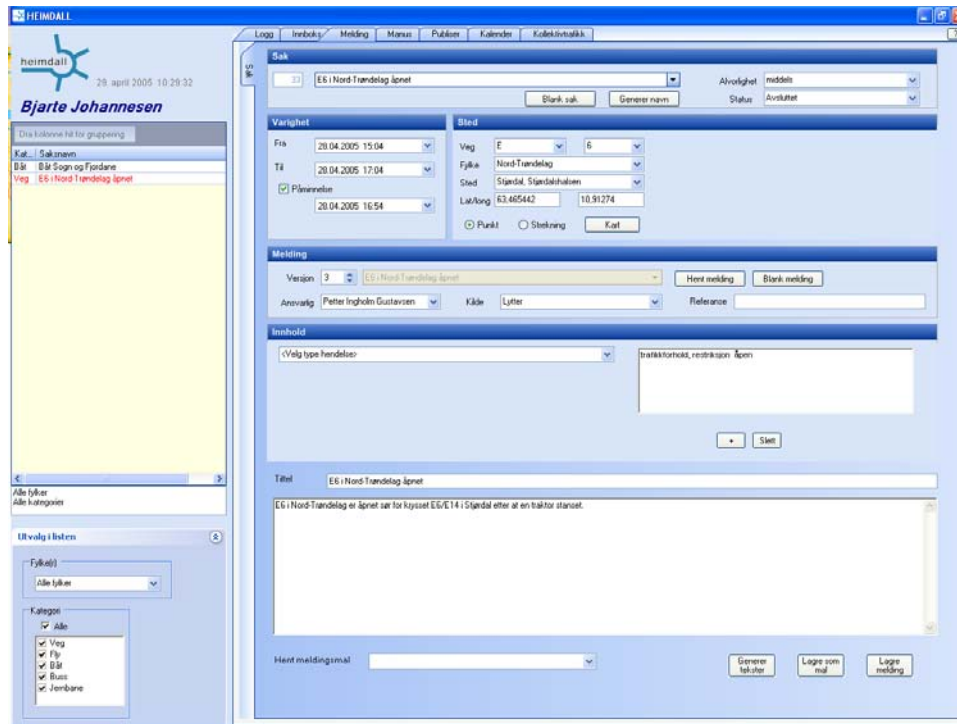


Figure 2.18: Screenshot from the Heimdall application

and the database is set to contain all roads, regardless of ownership and responsibility. Additional information such as the technical standard of the road, traffic signals and drainage will be present in the database. Information from distributed sensors in the road pavement will also be integrated, such as traffic density and noise and pollution levels. The system supports access control for the subsystems, and can be used to keep information confidential within government organizations. The database also supports information like accidents and speed limits for a road section.

NRK (by Roar Halten and Bjarte Johannesen) has provided information about their current internal application for handling traffic information. The application is named 'Heimdall', and screenshots are available in Figure 2.17, 2.18, 2.19 and 2.20. XML provides the foundation for the messages in the system, and they can be exported to and imported from TPEG compliant data sources. Events are associated with a set of latitude/longitude coordinates and can reference a particular road section. An event also contains information about the origin of the event (reported by a user, the road authorities, the police, etc.), and an optional description of the event. They are still implementing parts of the system, and interoperability with external message sources is one of the features planned for the future. See the figures for further information about features in Heimdall.

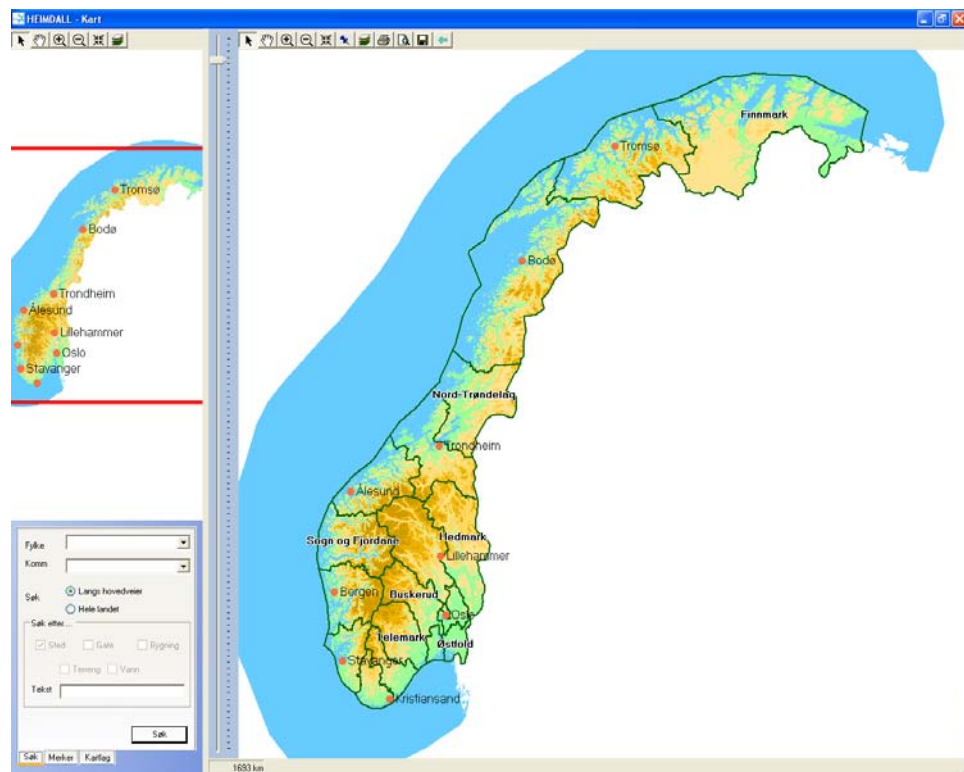


Figure 2.19: Screenshot from the Heimdall application

2.5.4 Available Commercial Car Navigation Solutions

Microsoft AutoRoute

Microsoft Corporation's [100] solution for mobile auto navigation in Europe is their AutoRoute [101] product. This product is available for laptops and mobile devices, such as PDAs and smart phones. The system is self contained and features a trip planning service and a map application. Features include:

- 800.000 points of interest
- 2.9 million miles of road
- Calculate fuel costs
- Drive-time zones (how far can I get in 1 hour?)
- Optimize routes and stops
- Drawing and annotation tools

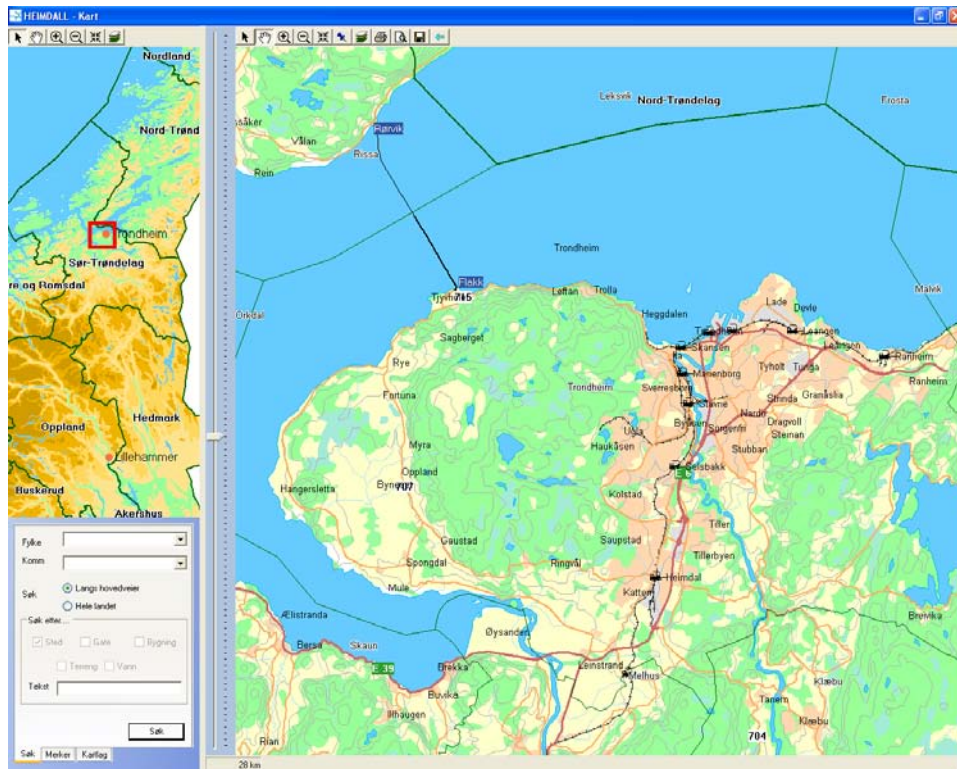


Figure 2.20: Screenshot from the Heimdall application

The last feature allows the user to create annotations and additional content in the map. This feature is local and doesn't use the networking capabilities of mobile devices. The product is primarily marketed towards the United Kingdom and Europe, and is built on Microsoft MapPoint [102] technology.

Microsoft Streets and Trips

While Microsoft AutoRoute is intended for the United Kingdom, Microsoft Streets and Trips is almost the identical application for the North American territory. People in the U.S.A. and Canada can use Microsoft Streets and Trips for navigation and trip planning. The feature set is similar to the AutoRoute product, but the North American version also features free online road construction updates and more points of interest.

Microsoft MapPoint

Although not developed for the mobile world, a branch of the MapPoint [102] suite is its programmable web services. These can be queried for context-sensitive information, which is relevant for the presentation on mobile devices. The MapPoint Web API features several services, including

directory lookup services ("find nearest"), a presentation service and a route calculation service. MapPoint is designed to extend and annotate existing georeferenced information. The application suite is a commercial product from Microsoft Corporation, but evaluation accounts are available upon request. A research project which use the MapPoint features is the GSM Cells project [56]. Other Microsoft products such as Microsoft AutoRoute [101], Microsoft Streets and Trips [103] and MSN Maps [104] also use MapPoint.

Route66 Mobile

Route 66 [105] was among the first commercially available route planning solutions on the Internet. They also built one of the most popular consumer applications for maps and route calculations. Their current product line is based on mobile applications, and include Route 66 Mobile 2006 [4]. Because the company is based in the Netherlands, their software is European centric and also has coverage in the Scandinavian region. Their feature set includes:

- Over 1.000.000 points of interest
- More than 7.700.000 KMs of road
- More than 76.100.000 house numbers (addresses)
- Coverage of Western Europe, 100% in central countries (Italy, Germany, Benelux, Switzerland, Austria and The UK mainland)
- 3 dimensional display interface
- Free, dynamic traffic information for the complete region

The traffic messages are delivered according to your location, but requires a network connection through the phone. Although the updates are free from Route 66, local phone carrier charges do still apply. The dynamic traffic information is also considered in route selection, and adjusts the path according to traffic information.

Tele Atlas GPS Navigation CD

The Tele Atlas GPS Navigation CD is a commercial product available from map data provider Tele Atlas, and is a pre-processed data set for car navigation systems. The data provided is a snapshot of the Tele Atlas database. Although Tele Atlas doesn't provide any car navigation system, their data are available in existing products on the market. Tele Atlas and NAVTEQ are examples of providers who have specialized in providing background information for mapping applications.

TomTom

TomTom [106] is a provider of commercial navigation software and services for mobile phones and PDAs. Mobile 5 is their most recent product for the mobile phone segment. They use a Bluetooth GPS to determine the current location, and the maps are supplied on a memory card. Mobile 5

supports downloadable content, but the main database is located on the included memory card. Their street network coverage is at 99% in most Western European countries. They also have software for PDAs and navigation solutions for both cars and motorcycles.

Wayfinder Navigator

Wayfinder Navigator [5] is together with WisePilot Navigator the only mentioned mobile phone product where you can use a navigation service without storing map data on your phone in advance. Their product include a Bluetooth GPS for lifetime (of the phone) navigation in Western Europe, online route planning and quarterly updates of map information. The Wayfinder Navigator supports traffic information, but while Route 66 provides this service for free, Wayfinder charges for the service. I have not been able to compare the two services, although they seem to be based on the same set of data (according to the countries the data is available for). The current data set for the Wayfinder Navigator includes 1.5 million points of interest.

WisePilot Navigator

WisePilot Navigator from Appello [107], or Telia Navigator as it will be known on the market, was launched by TeliaSonera [108] at the 23. of May 2005. Available as a subscription service for 3G customers, the application provides route calculations, downloadable content and coverage of Western Europe. According to a recent article [109], the service will have a monthly cost of around 80 Swedish kroner (8-10 euros, depending on exchange rates).

2.6 Other Standards of Interest

This subsection contains introductions to a few standards that did not fit into any other categories. These include standards for content exchange and presentation, and is further discussed in Chapter 4.

2.6.1 XML-RPC

XML-RPC [2] is an XML grammar tailored for the remote invocation of functions or methods over a transport carrier. Compared to traditional programming schemes where all application code is located on the same computer, Remote Procedure Calls (RPC) allow the invocation and the execution of the code to occur on separate systems. The client requests the invocation of a method with a set of parameters, the server executes the internal function and returns the result to the client. This enables the development of remote code libraries or Web services. RPC is handled transparent to the developer. XML-RPC uses HTTP as its transport method, sending messages defined in XML back and forth between the client and the server. Figure 2.21 shows the flow of data in an XML-RPC based method invocation.

An example of an XML-RPC message is included in Appendix A.4, and XML-RPC is also covered in Section 4.4.

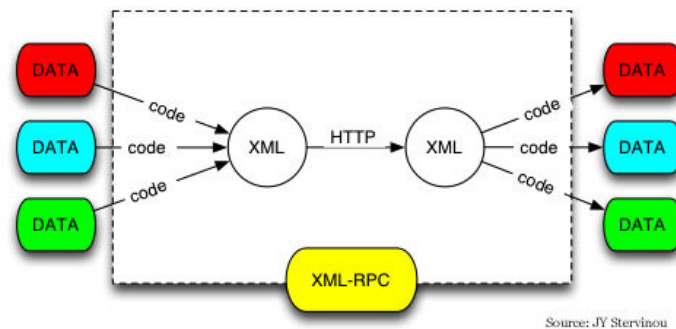


Figure 2.21: The flow of an invocation by XML-RPC (from [2])

2.6.2 Scalable Vector Graphics (SVG)

SVG [110] is an open standard for the presentation, animation and description of vector data. The standard is an XML based grammar, and several commercial applications support the format. Microsoft Visio [111] and Adobe Illustrator [112] are two applications with SVG support, and most users have SVG support through a plugin [113] developed by Adobe. This allows the presentation of SVG content in a web browser. Interactive applications may be developed entirely in SVG as the language supports ECMAScript [114] for procedural functions. Because an SVG document is also an XML document, the methods provided by programming languages to manipulate XML also applies to SVG. An example of a simple and basic SVG file is presented in Appendix A.3.1.

2.6.3 Really Simple Syndication (RSS)

The RSS 2.0 [6] specification is based on the Resource Description Framework (RDF) [73] and the RDF Site Summary (RSS) 1.0 [115]. RSS is a format for the programmatic exchange and aggregation of content channels. The RDF specification was included with the concept of The Semantic Web [37]. RSS compatible applications adjust the presentation method depending on context and user configuration. Version 2.0 of the standard incorporates metadata described in the Dublin Core (see Section 2.4.4) XML grammar, and is considered the de facto standard for aggregation of content on the Internet. The current standard is maintained by the Berkman Center for Internet & Society at Harvard Law School.

2.7 Conclusions

This background section has introduced the mobile platforms available for development. MIDP2 and Symbian C++ were both considered as platforms for this system. Experiences from previous projects indicate that the former is unsuitable because of the security limitations, while the latter features unfamiliar interfaces and harder debugging situations. Python for Series 60 is however

an interesting approach, and is frequently released in new and improved versions. The decision was therefor taken to go with Python for Series 60 for the prototype implementations. Coupled with the development of the server application also in Python, this creates an uniform platform across devices. The WMS standard is well suited for retrieval of background map data, and with the possibility of removing separate layers, our data may be super imposed without the previously recorded road data in the image. Other standards from the Open Geospatial Consortium (OGC) was also considered, but their field of operation was beyond the scope of this thesis. The standards are however well suited for this kind of project, because of their inherent support for interoperability across platforms. Integrating the Web Feature Service specification [36] for retrieval of historic features could be a future enhancement, in line with the already implemented choice of using WMS for background information.

Wikipedia has proved that user created content has much potential, and weblogs have become immensely popular. The content distribution model of weblogs still follow the tradition of storing information in several locations, and methods for creating informal network structures between content is still infant. The wiki distribution method is interesting, and if coupled with the most powerful tool for weblogs, the RSS stream, it can allow consumers to receive live feeds of relevant user created information. The Dublin Core metadata standard is also very interesting, and are already in use in the RSS 2.0 standard. Because of this, it was decided to include the RSS standard as on delivery mechanism in the project. SVG is available for many platforms, and since it's built on XML, easy to manipulate and generate programmatically. I therefor decided to go with SVG as an solution for exporting data from the repository.

Initial analysis indicates that there is a huge market for car navigation solutions, but that none implement user generated content in a satisfying manner. It would also be interesting to include support for user generated map data in a later version.

The background material is further discussed in Chapter 4.

Chapter 3

Scenarios

This chapter contains several imagined scenarios. These scenarios are included to help visualize and form a basis for the system described in this thesis. One of the cases are also explored further in Section 6.1, with a more detailed breakdown of the current situation in a Norwegian town. The inspiration for this thesis were formed by these scenarios, and the goal has been to explore the problems they highlight.

3.1 Informed About Traffic Conditions

Christer Edvartsen drives the same road each day to work. He carries an GPS receiver and his mobile phone with him at all times, and the position is constantly coordinated with a server. Three other people who decided to drive the same road that day uses the same system. Road work slows the traffic down to a grinding halt this day, and the system observes the fact that people are driving a lot slower than previous days. When a number of cars have been observed to move at a slower rate than usual, Christer gets a voice message on his mobile phone telling him that traffic ahead is slowing down. Although he seldom refrain from his routines, he decides to take the alternative route. Two minutes later than usual, but fifteen minutes ahead of his work mates, Christer arrives at work.

3.2 Manually Reporting Traffic Conditions

Christer drives the same same road to work, but again his attempt to reach work results in a delay. Being one of the first cars to be delayed, he decides to actively warn other people in the same area. He loudly says "report traffic jam", and his phone responds accordingly. The connection is made to the traffic center, and the area gets tagged as troublesome. Other persons in the vicinity are notified, in addition to people who are planning to use the given route within the next hour. They are able to select alternative paths, and although Christer is still stuck, he has been able to help other drivers. While the previous case section describes automatic extraction of meta data, this section annotates the previously extracted data with user supplied information.

In addition to the geographical information that we can extract for processing and mapping, this case illustrates some of the metadata that is available for applications. By using the temporal properties of the data, we are able to develop applications in other areas. These properties can be used in relation to dynamic data, such as traffic information, and to extract more static metadata, such as speed limits.

3.3 Annotating Traffic Information with Road Maps

While Christer has been driving to work each day with his traffic information system active in his car, the system has established a large set of points that are confirmed to be attributes of a road section. When Christer takes the car out for a ride on Sunday afternoon, a message arrives about a possible accident ahead. The message is annotated with a map, built from the points collected by drivers with the traffic information system installed, and features an almost real-time view of the road structure in the actual area. Figure 3.1 illustrates the situation.

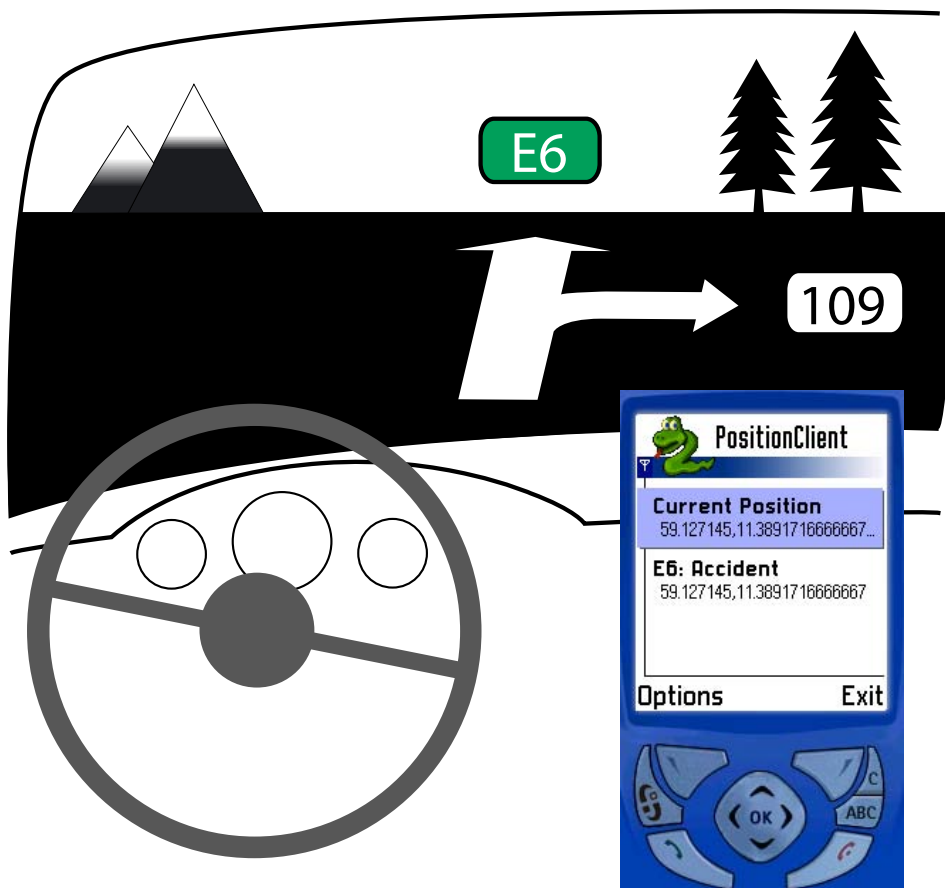


Figure 3.1: Scenario 3.3: Christer receives accident information

Although the data stored may not have a topological valid structure, the topology may be extracted at a later date. This will probably be the most feasible strategy, because it requires little user interaction when the points are collected. A "pure" topological solution would require more planning, both from the user and the application developers. Because mobile phones have limited resources, the amount of processing by an application should be kept to a minimum. To keep the interface as simple as possible, all topological information should be extracted from the data set on the storage server.

3.4 Dynamic Maps - Mobile Device

E6 is one of the busiest roads in Norway. The road evolves constantly, and this requires frequent shutdowns of road sections. This directs the traffic onto a new road, and the actual road used is different from the one stored in a map. When Christer are getting closer to a road maintenance area, his mobile display highlights the defunct section and presents a recent map of the area. Christer knows how to get back to the main road later, because of the included map image.

3.5 Dynamic Maps - Desktop Computer, Route Planning

While Christer is at home, he decides to plan his route for tomorrow. He brings up a web browser, and is able to see the current version of the road grid, modeled from the data collected by the clients that are on the road. The road network is under constant improvement and features other information reported from clients. Christer sees that the road he had planned to drive is closed, and decides to take an alternative route. Later he decides to drive to an unknown location where he has never been before. The map is recently updated, because someone else has driven past the same road segment recently. Figure 3.2 shows a client that has chosen another path, because of a closed road section.



Figure 3.2: Scenario 3.5: A road section is closed, and a client chooses another road

Chapter 4

System Design

The system described is a community based application where traffic information is provided from a number of swarming clients. Traffic situations exists in short intervals, and the overhead of each report should be kept small to retain usability. This indicates that current solutions are based on tradition instead of information value to the driver. Wikis (see Section 2.4.2) have received attention lately, and implementations like Wikipedia [67] has proved that the idea works on a large scale. By extending this idea into the area of geospatial and temporal information, mobile clients are able to report and modify events. The system is based on the ideas described in Section 2.4 regarding user generated content, in particular the wiki.

The first section of this chapter introduces the described system, its properties and how certain problems could be solved. In addition this chapter features five subsections, describing the available technologies and concepts that are used in this thesis. The field of interest include topics as the mobile devices used for processing, the positioning technology, the communication methods between the clients and the server, storage of client information and presentation to users in different formats.

4.1 System and Problem Description

The system described defines how to obtain, validate, distribute and present traffic related information. I describe the elements that I believe is unique for this system, and the reasoning behind including them in the problem description.

4.1.1 Obtaining Traffic and Event Information

The first problem in a traffic information system is to obtain information about situations that occur. Without being able to determine if a condition has been fulfilled or an event has happened, a traffic information is unable to provide any services. Current providers of traffic information (see Section 2.5.3) obtain most of their data from governmental sources, such as the Norwegian Road Authority and mass transit companies. Part of the information also stems from the public, who reports road conditions and events. According to P4 [97], members of the public usually report accident information before the authorities notifies the radio station. This indicates that the public are willing to report situations for the common good, and that they are doing so without a dedicated

incentive. The situation is that people want other to know about the queue they are stuck in, but have no means to do so. The first person to arrive at an event location, independent of the classification of that event, is always a member of the public. By allowing the public to report and share this information, it is possible to obtain almost instant notification of traffic related events.

Because many individuals carry a mobile phone wherever they go, this is our preferred platform for information collection. A description of the selected platform for the implementation is provided in Section 4.2. The system allows the users to store and update information as they please. This may seem like an uncontrollable situation, but projects (see Section 2.4.2) have shown that this approach is feasible, and can yield results of higher quality than commercial ventures.

This section indicates the need for mobile, handheld clients. These clients need to be able to report their current position, and should provide a mechanism to report that an event has occurred.

4.1.2 Automatic Retrieval of Traffic Information

By comparing the reported positions of a vehicle in movement, I am able to determine a set of metadata about the vehicle that may not be initially apparent. The movement of a vehicle will indicate the path of the road it is traveling along, while the distance and time between the points will indicate the velocity of the vehicle. This can be used to extract a set of properties for the road and for the vehicle. Classification of both can be performed, although an internal set of data may be as valuable. The programmatic value of such data is to detect variances and situations where the observed properties of a client deviate from the expected values. This information may include mean speed, acceleration patterns and rate of client observations. These properties describe traffic patterns and can be used to automatically generate information and traffic situation warnings. This process can be done independent of the driver or the client, and is based on the position of a client.

The collected information can be exported from the server to show the current state of traffic in a region. The range of what should be considered a deviation from the expected value is beyond the scope of this thesis, but a few estimates can be suggested. A classification depending on the amount of deviation could be divided according to a percentage relating to the average observed value of a given property. An example of a property is the mean velocity for an area, and if the current velocity of a vehicle remains at under 80% of the expected value through several observations, the area can be said to have reduced traffic capabilities. If we observe an even larger deviation from the expected values, such as a value of under 50% of the expected value, the conclusion may be that there is a potential for queues and slow traffic. Anything below 20% classified as queues, and the area should be avoided. The values may be adjusted according to other properties of the area.

A need for constant positional updates, enhanced by the extraction of metadata, and the storage of data sets for an area are properties of this subsection.

4.1.3 Validating Traffic and Event Information

The validation of traffic and event information is performed by the public. The public reports situations, and can enhance existing events. A person may report an event on the way to work, but two hours later, the situation has been resolved. The person deactivates the event or confirms that there's still a problem in the area. Because a client can be identified by a unique key, clients that

misbehave over time are put "on hold" or weighted as less reliable than other clients. The weight given to a particular report can be adjusted by the relationships between the clients, because one is more likely to trust information reported by personal friends than strangers. The idea is that because the public is the provider of the information, it should also be the entity that validates the data. This ensures that situations (such as queues) can be handled in a dynamic manner, and confirms the need for a version controlled storage system (discussed in Section 4.5), where it is possible to return an event to a previous state. A version controlled repository will also provide the possibility of presenting historical information. Figure 4.1 shows the lifetime of a message.

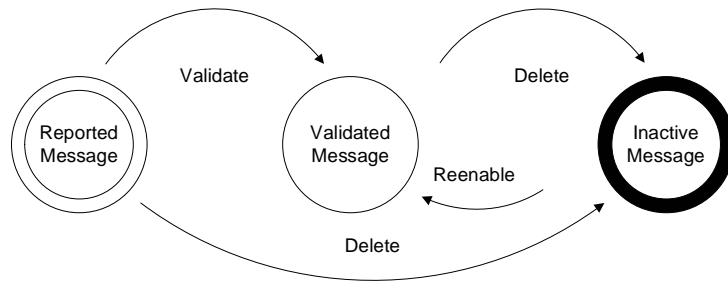


Figure 4.1: The lifetime and states of a message

This section specifies the need to be able to update, change and deactivate traffic information. Information should never be physically deleted, only marked as "not active". This would allow the information to be easily restored if deactivated for erroneous reasons.

4.1.4 Distributing Traffic and Event Information

The current practice (see Section 2.5.3) is to distribute information through the regular radio programming that people listen to in their car. The problem with this system is that it is impossible to limit content by location. Some commercial entities attempt to distribute traffic information and traffic messages in a digital format (see Section 2.5.4). NRK incorporates TPEG (see Section 2.5.2) support into their recently launched application (see Section 2.5.3) for handling traffic information, but there is currently no public distribution of the TPEG stream. The idea is to distribute the information only to those clients which are in the affected area. Because each client is able to determine its position (further discussed in Section 4.3), it may request information that is relevant to its current location. Two methods of distribution are interesting in the context of traffic information, and are discussed in Section 4.4. The clients could receive a notification when something occurs in their close vicinity, or they could check in with the server at regular intervals.

4.1.5 Presenting Traffic and Event Information

The presentation of the information is one of the difficult tasks of this thesis. A lot of research and development has been invested in user interface design, but I will not mention anyone beyond the scope of Section 2.5.1. The idea is to provide the information in a application independent

format, so that different services can be implemented. Presentation services can be built on top of the communication layer (see Section 4.4) to avoid tight integration with other parts of the system. Mobile devices present in vehicles should not disturb the driver or the operation of the vehicle. Traffic safety is the most important issue, and information must be presented in a clear and non-disturbing manner. Other situations may require a more informal presentation, such as presentation on a desktop computer. The user may want to perform tasks such as route planning, navigational aids and map lookups, where a minimal interface may be unsuitable. The user's focus can also be dedicated to the application, in contrast to a car-based situation.

This section illustrates the need for an presentation independent method of data exchange, where several different presentation modules can be developed. This is further discussed in Section 4.6.

4.2 Mobile Devices

The area of mobile devices provides several suitable units for a project like this. The decision was taken to go for the current generation of high end mobile phones, supporting the UMTS 3G (see Section 2.1.1) standard. This allows for good data transfer rates in urban areas of Norway, in addition to providing lower quality communication channels in rural areas. Communication is an important aspect of this project, and other device classes would have to be extended with an external communication device to communicate with the server. If this project was to be performed three years ago, a laptop based or PDA based solution would have been the choice, but because the capabilities and available developer tools for mobile phones has increased greatly since then, the current generation provides the required functionality.

The chosen device is the Nokia 6630, a 3G capable phone featuring a 1.3 megapixel camera, the Symbian 8.0a operating system (see Section 2.1.2) and the Series 60 (see Section 2.1.2) platform. The device is fairly high end, provides a good amount of processing power, extensive application support and changeable memory cards in a several sizes (based on the MMC-standard [116]).

4.2.1 Application Development

Because the focus of this thesis is to develop a large scale solution for reporting and updating traffic information, a common platform is a necessity. The initial plan was to use the Java MIDP2 environment (see Section 2.1.2). MIDP2 is the largest common profile for mobile applications, and features several optional modules. These modules contains interfaces to APIs that are interesting for this project, such as Bluetooth (see Section 2.1.2), file access and IP-based communication. The severe security limitations of the Java virtual machine, and because most devices don't allow the user to disable these limitations, the development of non-interactive applications under MIDP2 is impossible for other purposes than simple applications.

Previous projects have explored the possibility of developing a solution for the C++ interface of the Symbian OS (see Section 2.1.2). Applications that are developed for Symbian runs in a relaxed security setting, and allows the developer full access to the environment. This solves the problems caused by the security limitations of the Java platform, but introduces its own problems. Because of the tighter integration to the operating system, the application has more responsibility. Memory management, a lack of documentation and examples, unfamiliar interfaces and harder debugging

situations are areas that suggest another development platform than native C++. The amount of Symbian based phones on the market is acceptable, although their market share is smaller than MIDP2.

Another Symbian based solution is Python for Series 60 (see Section 2.1.2). This solution brings the native power of the Symbian operating system to a simpler environment than C++. By installing the Python virtual machine on a Series 60 capable phone, the developer is able to access native functionality of the operating system, but still has a high level language for the development of applications. The Python language [21] has proven to be of high quality and has a large and active community supporting it. This combination of low level details from the operating system and the support for a high level scripting language allows for the rapid development of prototype applications. The extensive set of Python modules and extensions are also available for the mobile version of the language, and other 3rd party extensions can be installed by transferring them to the phone by Bluetooth [22].

4.3 Positioning

Since I am interested in the position of the individual clients, I need a way determine their position. This process must be automatic and performed at regular intervals, and a degree of accuracy is essential. Several different positioning technologies have been evaluated, although the services provided by mobile service operators are outside the scope of this thesis. The reason for this is that these services are not available across borders, operators or network types. The OpenLS standard was also considered for implementation, but the scope of the specification is too extensive for this project.

The selected positional technology is the most used position technology available, the Global Positioning System (GPS). Based on a network of satellites orbiting the earth, a GPS-receiver determines its position by calculating the time-difference between the satellites. The GPS technology is accurate down to 50 meters, but can be enhanced by using a differential GPS. The default accuracy of the GPS is enough for this project, and because I want to use consumer grade hardware, GPS was my choice. To obtain the position from GPS I use a Bluetooth-enabled (see Section 2.1.2) GPS receiver that reports the current position and sets of metadata, such as satellite coverage, dilution of precision, speed above ground and current heading. The data is delivered as NMEA-formatted (see Section 2.2.4) strings, and can be both logged and parsed directly on the mobile device. GPS hardware is cheap and provides good accuracy compared to the investment.

Less accurate positioning methods are also available. These include a user-supplied textual identifier describing the location, or using the current cell tower id as an identifier. Both methods allows phones that are not equipped with an GPS to report their position, and are available on existing phones. Information about the current cell tower is available in Python for Series 60, and can be used to determine an approximate distance to an event. The cell tower id is in itself not of any value, but coupled with a cell tower id database, one is able to resolve the id to a set of coordinates. There are a few projects available that attempt to build such as a database [56]. A textual identifier can be resolved through a gazetteer service, a database that contains mappings from textual identifiers to geographical coordinates. A few such services exist today, but none contain

local names spanning the globe. Due to the immature status of the current cell tower databases and the unavailability of a free data source for a global gazetteer service, the decision was made to base the project on collected GPS data.

The positional module in a given application should remain easy to replace and self-contained under any circumstances. This is necessary to ensure the possibility of exchanging the module with a new positioning technology at a later date, or to allow for exchanging positional services during runtime. A gazetteer or cell tower id service can be added at a later time as a new module.

4.4 Communication

The idea of distributed clients requires communication between clients. There are two possible concepts for communication in this project, the peer to peer structure and the client/server approach. While it may be tempting to see the clients as individual units without any dependency of a server, it is hard to implement in practise. Because we require concise and persistent information to store event and client information, a peer to peer network will be hard to implement. Even if we ignore the communication overhead present in a peer to peer system, we are left with a larger implementation footprint than with the client/server approach. There are several libraries available to help with the development of peer to peer applications, for example the JXTA [?] framework for Java. Some research in the area of traffic information has also explored a peer to peer network structure (see Section 2.5.1). The client/server architecture is a more suitable solution for this project, as it is easier to implement and the storage of persistent data is vital.

Even in a client/server architecture a protocol for communication is needed. The options are to either develop a protocol or to implement an existing one. The former allows for a custom, well-suited protocol, but requires more development time. Extendability is harmed, and 3rd party implementations suffer because of the non-standard communication method. When considering these issues, it is apparent that the development of a custom protocol should be avoided. After investigating the available options, including SOAP, CORBA and other protocols for programmatic information exchange, XML-RPC (see Section 2.6.1) was the best option. Offering a lightweight, easy to read and understand remote procedure call protocol, XML-RPC provides remote invocation of server methods. Because XML-RPC is based on XML, the messages between the server and the clients are passed along as XML documents. This means that languages that has XML support, usually also have an XML-RPC implementation available. This language independency proved valuable in development, because different modules could be developed in different programming languages.

The XML-RPC server exposes a set of methods to clients, which they can invoke with a set of available parameters. XML-RPC is only concerned with the reception of arguments for a function and returning the result of function execution. Many languages have an XML-RPC implementation that exports internal objects and functions and provides a framework for handling connections. Because XML-RPC uses HTTP as its transport protocol, proxy, authentication and encryption (HTTPS) services are still valid for XML-RPC. This allows further refinement of the protocol abilities, by using existing tools for the HTTP protocol. These tools also include non-traditional items (in a programmatic context) such as caching and automatic indexing of remote methods.

There are several frameworks described for the publishing of remote processing capabilities, such as WDDX [117], which can be used to obtain information about remote services.

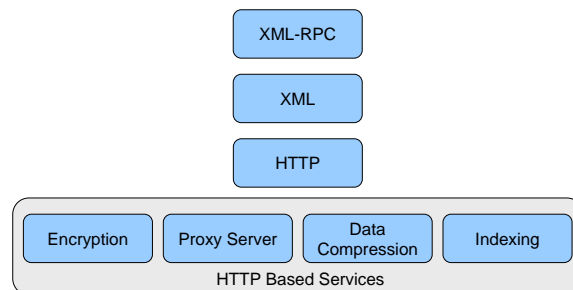


Figure 4.2: XML-RPC and the advantage of using HTTP as a transport protocol

XML-RPC comes with a price: verbosity. The transfer of verbose information costs more (in processing, used air time and as a monetary cost) than native information exchange formats. The format can however be parsed independent of programming language and platform, which helps to avoid issues such as endian problems [118]. The resulting HTTP stream can be compressed (as it is textual information) to reduce the transfer cost, which is probably the most problematic issue for participants. This can be adjusted on a per-request basis, where the larger requests are compressed to save air time. XML-RPC became the choice for this application because of its simplicity and independence of a particular programming language. This became an advantage in the implementation phase (Chapter 5).

OpenLS (as described in Section 2.3.3) is an open standard intended for the development of location based services. While this standard appears to be well-suited for exactly this project, the implementations are few and the standard extensive. This thesis is only considered with a subset of the core services described in the standard, and a complete OpenLS implementation would require a larger project. Some of the possible methods described in Section 4.3 could contribute to an OpenLS based implementation.

4.5 Storage and Transfer Formats

As described in Section 4.4 XML-RPC was the chosen method for client/server interaction. The communication between the client and the server is performed by sending small XML messages. Other transfer formats are mentioned briefly in Section 2.2.1 and Section 2.2.3. These formats were developed for the exchange of geographical information, and has potential for a project like this. While the GML3.0 format is extensive, both GML3.0 and GPX lists suitable features. The "speed above ground" and "heading" properties are missing in the current version of GPX, so a GPX based solution would have to use a previous version. Both specifications are open standards freely available as XML grammars. Nevertheless, I decided to use XML-RPC, because this makes the implementation phase easier, and the standard is implemented across platforms.

An important criteria in the described system is the need for version controlled and historic

information. The repository for the collected information must be able to go back to any instant in time, and extract the information available at that moment. This allows us to extend the system to provide time based analysis of the traffic situation, of the sequence of events and to recover historic information. A suggested specification for such a storage system has been developed by the author and two colleagues [119]. The specification describes a repository for storing version controlled information, without requiring a new version of a feature to be stored in its entirety. Version controlling the repository at an event or client level allows us to retrieve historical data, as well as return individual events or clients to a previous state. The repository is based on GML3.0 and is an implementation specification. This thesis is however limited to a smaller set of historical information.

4.6 Presentation

Without the ability to extract useful information from the current state of the repository, users will not be able to use the system. As specified in Section 4.1 an independent protocol for data exchange makes it possible to add new presentation services to the information engine. With the description provided in Section 4.4, this property is preserved and the layer separation is maintained. The details of a presentation module is contained within the module, and the modification of one module does not affect other presentation modules. This allows us to create presentation services for several formats, a few which are described in the implementation chapter (see Chapter 5).

Several standards should be included in an implementation, the Web Map Service (see Section 2.2.2), Scalable Vector Graphics (see Section 2.6.2) and Really Simple Syndication (RSS) (see Section 2.6.3). The first provides background imaging such as rivers and city boundaries to further annotate the information presented from the traffic information system. SVG is concerned with the presentation of graphical objects to the viewer. Objects include images, lines, paths and dynamic content. A complete application can be built with SVG, which is also available on mobile devices [120]. The third specification, RSS, is an XML markup that describes a content delivery channel. The channel can contain arbitrary information, and can be used for the delivery of traffic information and events. Further refinement and filtering can be applied by the RSS presentation module. RSS readers are available en masse for desktop computers, and is the industry standard for the syndication and aggregation of content. A recently developed initiative is the TPEG standard, which is described in Section 2.5.2. A presentation module may provide information according to the TPEG specification, which can be used by conforming clients.

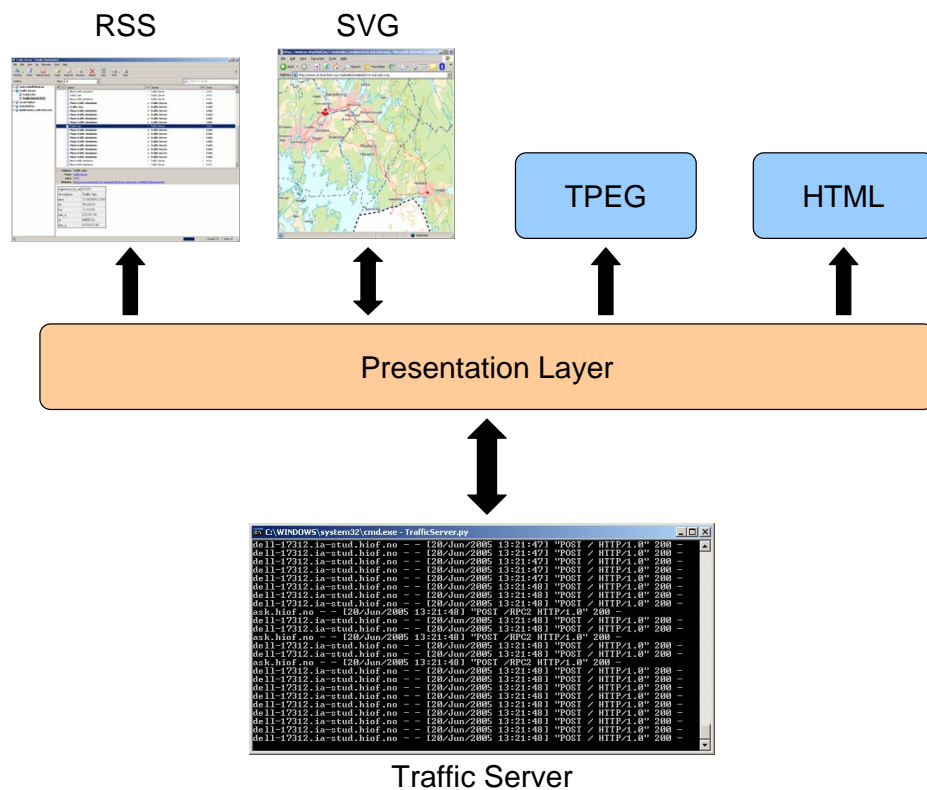


Figure 4.3: The presentation layer provides several output formats to clients

```

- <rss version="2.0">
- <channel>
  <title>Traffic Server</title>
  <link>
    http://www.ia-stud.hiof.no/~matsslin/traffic/index.php?rss=all
  </link>
  <description>
    An interactive, georeferenced traffic information service built after the principles of collaborati
    for a given area. Use ?rss=all to retrieve all registered events, regardless of location.
  </description>
  <language>en-us</language>
  <item>
    <title>Minor traffic slowdown</title>
    <link>
      http://www.ia-stud.hiof.no/~matsslin/traffic/index.php?event_id=Y0zlk3tt&format=html
    </link>
    <description>Minor traffic slowdown</description>
    <dc:Author>31337</dc:Author>
    <dc:Date>Thu, 23 Jun 2005 12:00:27 +0200</dc:Date>
    <dc:Identifier>Y0zlk3tt</dc:Identifier>
    <dc:Coverage>11.02009.59.259731666667</dc:Coverage>
    <dc:Description>Minor traffic slowdown</dc:Description>
    <dc:Source>Reported</dc:Source>
    <dc:Rights>Public Domain</dc:Rights>
  </item>

```

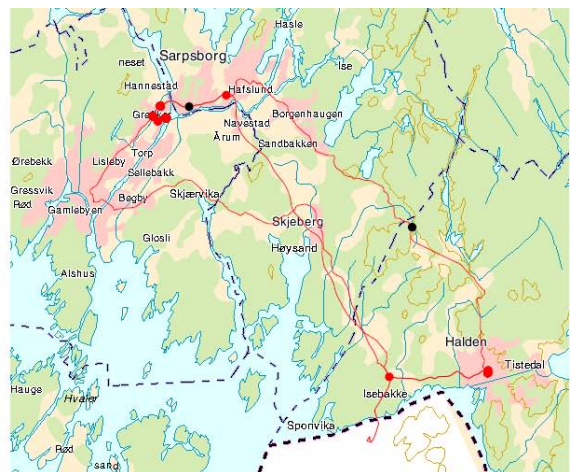


Figure 4.4: RSS and SVG representations of data

Chapter 5

Prototype Implementation

This chapter introduces a prototype implementation of the project described in Chapter 1 and Chapter 4. The implementation choices and particular details are covered in this chapter. The results and future work are discussed in Chapter 7. An overview of the implementation begins the chapter, before the subsystem implementations are introduced.

5.1 Implementation Overview

The implementation phase was conducted to investigate the protocols, technologies and concepts introduced in Chapter 4. Three subsystems are included, and their implementation details are separate from the other modules (see Figure 5.1).

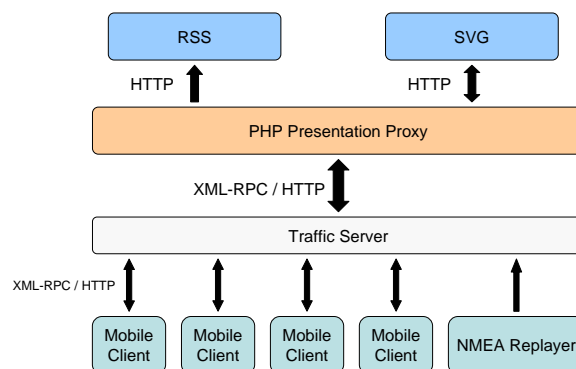


Figure 5.1: Architectural overview

The implementation consists of three different modules:

Client

The clients are the contributors to information to the system. Based on mobile phones running the Series 60 platform, the clients are powered by Python for Series 60. The clients connect to the Traffic Server through XML-RPC, and update the server with their position in regular intervals. An application to simulate and play back captured client information is also included, and discussed in Section 5.5. The mobile client is based on the scenario described in Section 3.2 and may incorporate information as described in Scenario 3.4 at a later stage.

Server

The Traffic Server maintains connectivity between the different clients and their history. Event handling is also performed in the server module, and it's also responsible for detecting and generating automatic event notifications. The historic information of the clients are available to the other modules. The Traffic Server was written in Python and interaction with the server is done through XML-RPC (see Section 2.6.1). A scenario describing these features is described in Section 3.1.

Presentation

The presentation module is responsible for presenting the collected information to the user in a useful manner. A proxy layer is introduced between the Traffic Server and the information display. The service was written using PHP [121], and communicates with the Traffic Server through XML-RPC. The implementation for displaying information uses SVG (see Section 2.6.2) to present the data over the Web. The background imaging is provided through WMS (see Section 2.2.2), and the road information is retrieved from the Traffic Server. The system also produces an RSS (see Section 2.6.3) stream containing event information. This stream can be imported by RSS compatible readers, and the presentation is handled by the RSS client. This subsystem implements parts of the functionality described in Scenario 3.3 and Scenario 3.5.

5.2 Mobile Device Client

The mobile clients are the terminals used by people in the field. Reports are submitted from the mobile clients, and they receive notifications when new events are registered. The clients are developed in Python for Series 60 (see Section 2.1.2) and uses XML-RPC for communicating with the Traffic Server.

The initial plan was to develop separate applications for a set of pre-classified events, and then use voice commands to report an accident or traffic problems. Because of limitations in the software on the Nokia 6630, it was not possible to associate a 3rd party application with voice activation. The mobile client was therefor developed as one application, and is responsible for updating the position of the client, registering events and deleting events. The location is determined by a Bluetooth GPS, which delivers NMEA compliant strings through a Bluetooth serial interface. The application detects Bluetooth devices in the vicinity, and lets the user choose the correct one. This sequence is implemented by the Python for Series 60 framework (which uses the underlying Symbian OS), and

returns a tuple describing the selected Bluetooth device. Figure 5.2 shows the mobile client running in an emulator, and displays the interaction required to add a new event. The events location is determined by the location values supplied by the Bluetooth GPS.



Figure 5.2: The Client: Adding a new event in the field

The application communicates with the Traffic Server through XML-RPC over GPRS or if available, the UMTS 3G network. At startup the application registers with the server, and receives a unique session key. The key is stored internally, and used for each subsequent request to the server. The request most often performed is `UpdateRequest`, and is used by the client to submit its position and velocity to the server. The server locates new events in the vicinity of the new position, and returns any new information. The events are only transferred once, and the client will not receive any new notifications of the event unless the details change. A list containing the close events are the main interface of the application, and their description is shown together with the reported position. A reverse geocode service would allow the client to display a more suitable reference to the location, but because of the unavailability of a free service for such requests, this was not implemented.

The clients can manipulate events as they see fit, and their actions are replicated on the Traffic Server. Figure 5.3 shows how the user is able to delete and reconfirm reported events. The application doesn't supply any additional information, such as map details of the area, but the development version of Python for Series 60 indicates that better bitmap support is coming in the next version.

5.3 Traffic Server

The Traffic Server is the administrative unit in the prototype implementation, and is written in Python. Client coordination, message distribution and simple geospatial analysis are the responsibilities of this module. The Traffic Server offers services to clients and the presentation proxy.



Figure 5.3: The Client: Deleting the new event

These services range from coordinate updates to data retrieval queries. The server interface is available through XML-RPC, and the server exports its available methods to clients. Because clients are required to authenticate with the server before requesting a service, a client has to log in to the server. This is done by invoking the `RegisterClient` method, and passing an username and a password as the required parameters. Authentication is then performed, and if successful, a secret session key is returned. This session key is then used for subsequent requests, and uniquely identifies the client in question.

The request pattern after the initial authentication depends on the type of client. A field client will probably remain active for an extensive time and will perform several queries. These queries will mostly be `UpdatePosition` queries, which updates the current position of the client and returns any events found to be within the close physical area (20km in the implementation). The clients are able to manipulate the current data structures through these methods, but some integrity checking is maintained to ensure data consistency. Methods responsible for validating a session key and a coordinate pair are also available. For a complete reference to the available services from the Traffic Server, please refer to Appendix B.1.

The Traffic Server is also responsible for storing the historical values of events and clients. This information can be retrieved by the presentation layer and the clients, and an example of this feature is shown in Figure 5.10. Historic event information is also an important property, as this allows a client to re-activate events that have been closed erroneously. It also allows us to examine the flow of events and clients in hindsight. The Traffic Server supports very limited geographical filtering of events based on the bounding box of the target area. As discussed briefly in Section 4.5 the current storage solution used in Traffic Server is an internal list of events and clients. These are manipulated as the server receives and generates events, but the content is not stored between server sessions.


```

C:\WINDOWS\system32\cmd.exe - TrafficServer.py
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:47] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:47] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:47] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:47] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:47] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
ask.hiof.no - - [20/Jun/2005 13:21:48] "POST /RPC2 HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
ask.hiof.no - - [20/Jun/2005 13:21:48] "POST /RPC2 HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
ask.hiof.no - - [20/Jun/2005 13:21:48] "POST /RPC2 HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -
dell-17312.ia-stud.hiof.no - - [20/Jun/2005 13:21:48] "POST / HTTP/1.0" 200 -

```

Figure 5.4: Traffic Server XML-RPC log

This means that when the server restarts, it starts with an empty repository. A suggested storage solution is described in the OneMap Repository [119], and specifies a version controlled, XML-based repository for geographical features. Other options are open source alternatives such as MySQL Spatial Extensions and PostGIS. These have not been evaluated, and the current implementation has proved complete enough for prototype testing.

Automatic generation of events is the responsibility of the Traffic Server. When a client submits a new location, the coordinates are snapped to the closest intersection in a grid. This grid value identifies the particular point. The grid size in the prototype implementation is based on 20 meter intervals. This approach was sufficient for simple detection of mean speed deviation in an local area, but must be extended or interpolated if queries are performed at 30 second intervals. The coordinates are alone not enough to correctly identify a road section, because bridges and tunnels complicate the road network. This was solved by basing the hash table lookup on three values, in particular the x location, the y location and the heading of a vehicle. These three values formed a tuple (x,y,heading), and was then used as an index in the table. The heading value is identified by one of sixteen section values, each corresponding to a given section of the unit circle (see Figure 5.5). If a point has been visited earlier, the current speed is compared to the expected speed. If a level of deviation is detected (see Section 4.1.2), an event is generated at the location. The memory requirements for storing the grid is minimized by using a hash table, because this allows me to only store locations that have observations.

5.4 Presentation

The presentation layer is implemented as a presentation proxy written in PHP [121]. This acts as an interface to the Traffic Server for the SVG and RSS presentation services. The presentation proxy communicates with the Traffic Server using XML-RPC and converts the information to a more

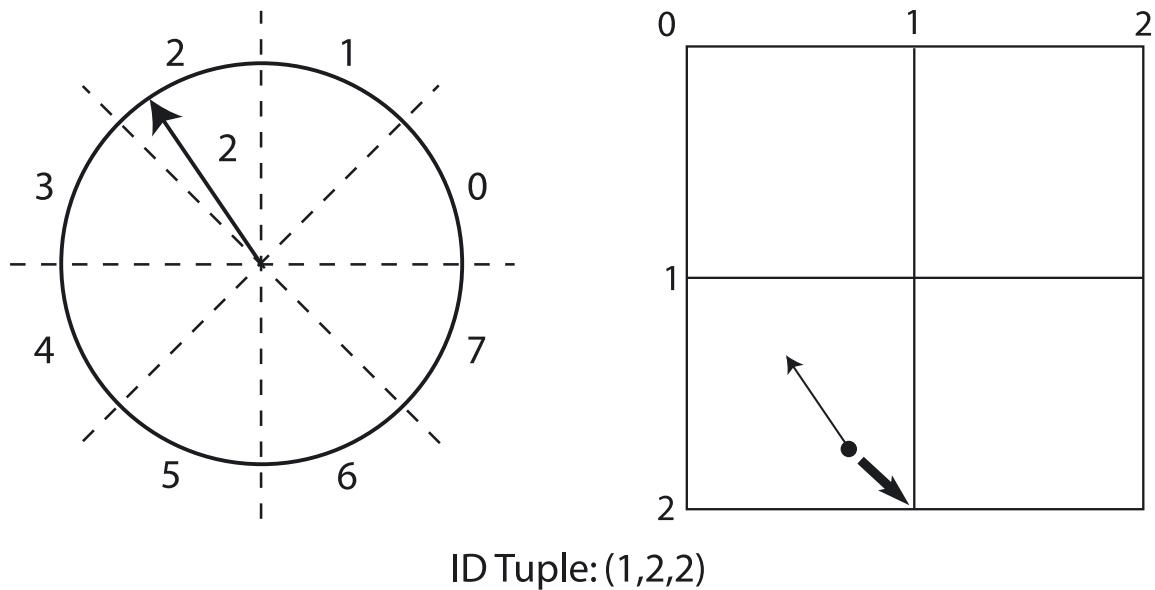


Figure 5.5: Grid index id is determined by snapping the location to the grid, and selecting the correct section for the heading

suitable format for the two presentation modules.

5.4.1 SVG Interface

The SVG Interface is the main method of interacting with the Traffic Server framework through the World Wide Web. Scalable Vector Graphics (see Section 2.6.2) is a W3C [54] recommendation that specifies an XML grammar for vector graphics and user interface events. The SVG Interface in the prototype is separated from the Traffic Server by an in-between web service written in PHP. The SVG is downloaded to the client and viewed directly in a web browser (with SVG support). Because SVG supports ECMAScript [114] as a native language for event processing, procedural code can create a dynamic application.

The SVG Interface has three responsibilities. It should provide a dynamically created backdrop for visual reference, it should display active clients on the map and it should show events that have been reported. The first objective is satisfied by using a dynamic WMS (see Section 2.2.2) service in addition to the historical data available from the Traffic Server. The second and third objectives are satisfied in the cooperation with the presentation web service. This service is separate from the SVG Interface, and the SVG Interface accesses the PHP server at regular intervals requesting updated information. The information is not delivered directly to the client, but the client requests the information at 30 second intervals. This property could be enhanced by developing a push based application, but the security and technical limitations of ECMAScript running in an SVG document may prove troublesome. Figure 5.7 shows one active client in the lower region of Østfold, while

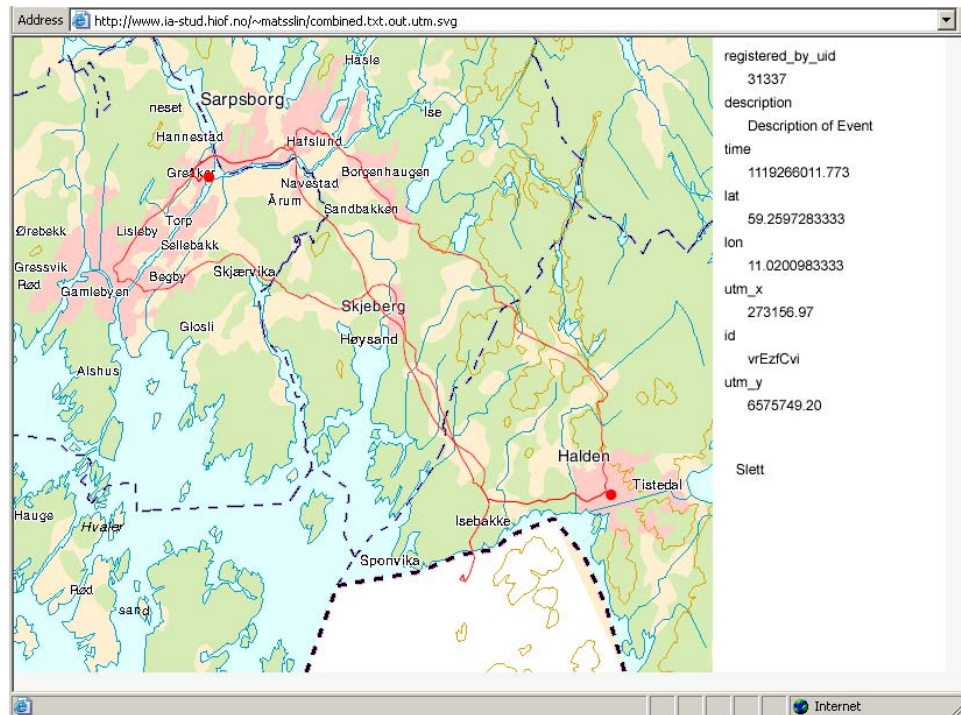


Figure 5.6: SVG Interface: Viewing an event and its details

Figure 5.8 displays several clients in the same area. The black dots are clients, the red indicates that an event has occurred in the area. The user may request more information about both the event or the client by moving their mouse pointer over the symbol.

The technical implementation of the SVG Interface is done using the mentioned combination of ECMAScript and a server side PHP application. Figure 5.9 depicts the information flow performed at 30 second intervals, where the SVG application makes a HTTP based connection to the PHP server, downloads new SVG content and incorporates it into the existing SVG structure. The physical SVG file only provides a set of empty XML elements representing the visual layers. These elements are repopulated by ECMAScript at thirty second intervals. The new content is downloaded from the server, and using the `parseXML()` function in the Adobe SVG Viewer, the new content is imported into the displayed SVG document. This dynamic feature create interactive SVG applications with Web based content. When the mouse pointer hovers over a symbol featuring more information, the information is requested from the PHP application. The console element in the SVG structure is populated with the information, and is displayed on the right side of the user interface. In the prototype this is the unprocessed information reported by clients, or generated by the traffic server, but a commercial implementation would perform data sanitation and feature a more visually pleasing presentation. This is however not the objective of this prototype, and the properties are included to show the technical details.

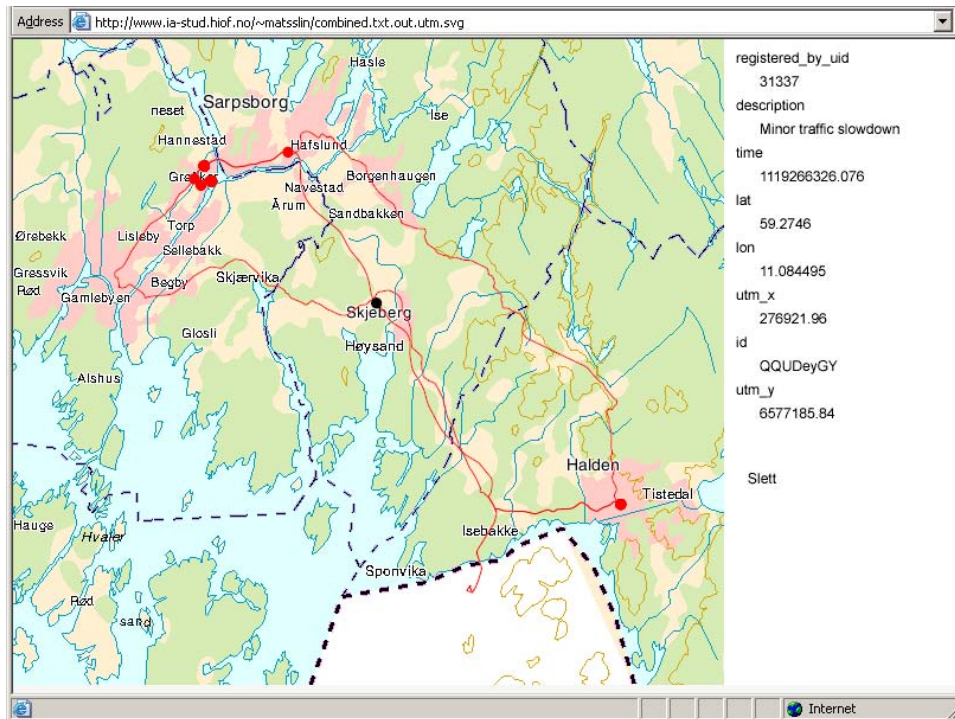


Figure 5.7: SVG Interface: Viewing an event and one active client in the map

The initial SVG is produced by an application named SVGGenerator (written in Python). SVGGenerator connects to the Traffic Server and requests all historical information. This information is processed, and an SVG document is generated. The SVG features an URL to a Web Map Service (see Section 2.2.2) for background imaging, and superimposes the historical point data as road networks. The result contains all the information one would expect from a mapping solution, and is presented in Figure 5.10. The view is scaled according to the geographical area featured in the historical data set, and data sets with a smaller distribution of points will be more local in area. Figure 5.11 and Figure 5.12 illustrates two local point sets. The important issue here is that these maps are built from scratch by the information available in the historical repository, and does not feature any initial road data. These images are also further enhanced by dynamic event and client information. The current prototype doesn't adjust the map size or regenerate the background imaging when clients move into unmapped geographical areas, but this could be added later. Some experiments were also performed with the type of display, and Figure 5.13 shows the road network weighted by average velocity at the location.

5.4.2 RSS Interface

The RSS Interface delivers data in the format described by the Really Simple Syndication standard (see Section 2.6.3). This format is well-suited for content aggregation services, and for applications

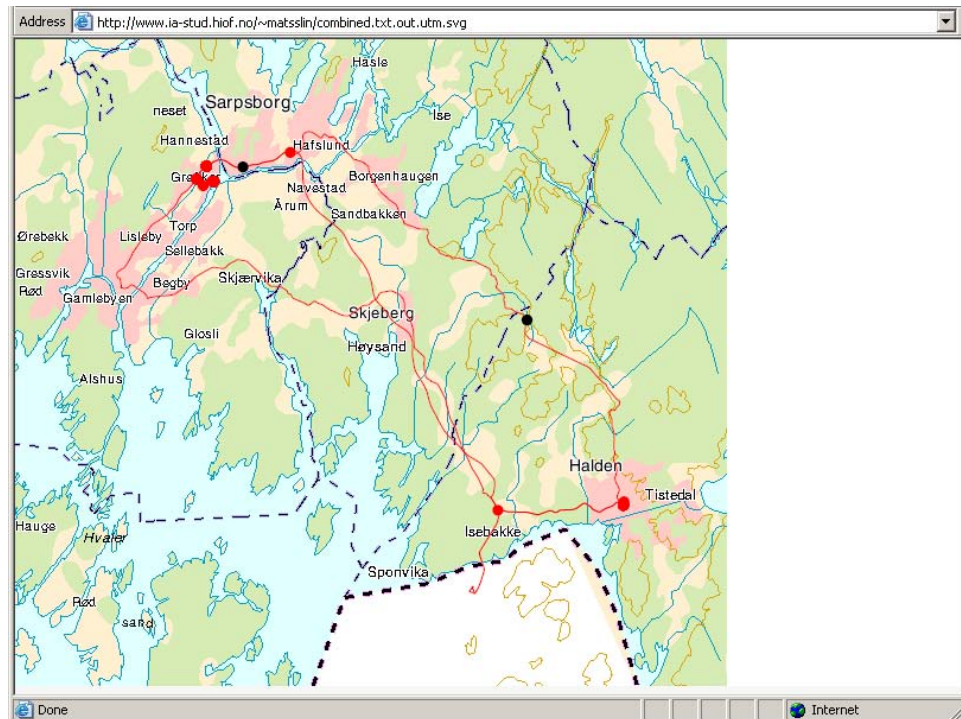


Figure 5.8: SVG Interface: Viewing several events and active clients in the map

that need programmatic access to the contents of a resource. An RSS capable server may deliver one or several channels of information, where a channel may contain a specific subset of the available data. The RSS interface in this project delivers one channel containing a set of traffic messages, possibly filtered by parameters provided by the user. By supplying a bounding box area, the user is able to generate an RSS stream that contain the georeferenced traffic information of particular interest. This stream can be added to any application that supports RSS channels.

The implemented solution uses the RSS 2.0 standard, and enhances the content with metadata following the Dublin Core (see Section 2.4.4) grammar. The metadata includes location, a description of the event that has occurred, when it occurred and who reported it. A link to more information about the event is also provided. The resulting page from a request for more information may include a small map of the area, a historic overview of the location, information about clients that are in the area and similar potentially interesting information. An example of a live RSS stream is given in Appendix A.2.

The RSS stream is produced by a small PHP application. The PHP application acts as a web server to external clients, but connects to the Traffic Server in the background through XML-RPC. It fetches the information needed to provide an RSS stream and then closes the connection. The output is formatted according to the RSS 2.0 specification, and returned to the requesting client. An RSS client can then display the information in a suitable way, one example being how Mozilla Thunderbird (see Figure 5.14) presents the information as regular E-mail documents. This allows

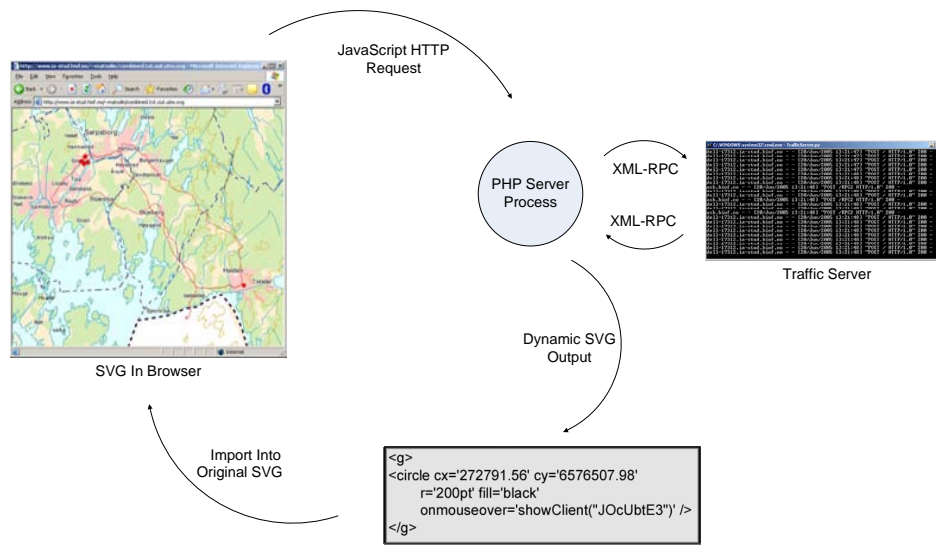


Figure 5.9: SVG Interface: Flow schematic for dynamic SVG

the user to get an automatic notification each time an event occurs in their selected area.

5.5 NMEAREplayer

The NMEAREplayer is an application developed to simulate a mobile client. The application takes a file containing NMEA strings as input and connects to the Traffic Server. It then simulates a mobile client by reading the NMEA strings and performing the associated actions. Coupled with a small debug application that can be run on the mobile device, this allows a developer to log and replay an GPS session. Because the application uses the same data each time, the application state can be replicated across debug sessions. The file contains information about which satellites were active, the horizontal and vertical dilution, the position of the client and other available data. The application also accepts a parameter to modify the speed of which the data is repeated, making it possible to replay a session faster than it was recorded. This allows me to simulate several clients in a sand box environment, where I'm able to add new clients to the system by starting a new process.

The NMEAREplayer only reports the location of the vehicle. This is the only information that is reported to the traffic server in the current system, but can be extended by editing the application. The application only considers locations where the GPS receiver has a location fix. The delay between each update is calculated as follows:

```
time_to_wait = time_passed_in_nmea / ratio
```

The time I use to process the NMEA-stream and calculate the current time is then subtracted:

```
real_time_to_wait = time_to_wait - (real_time - previous_real_time)
```

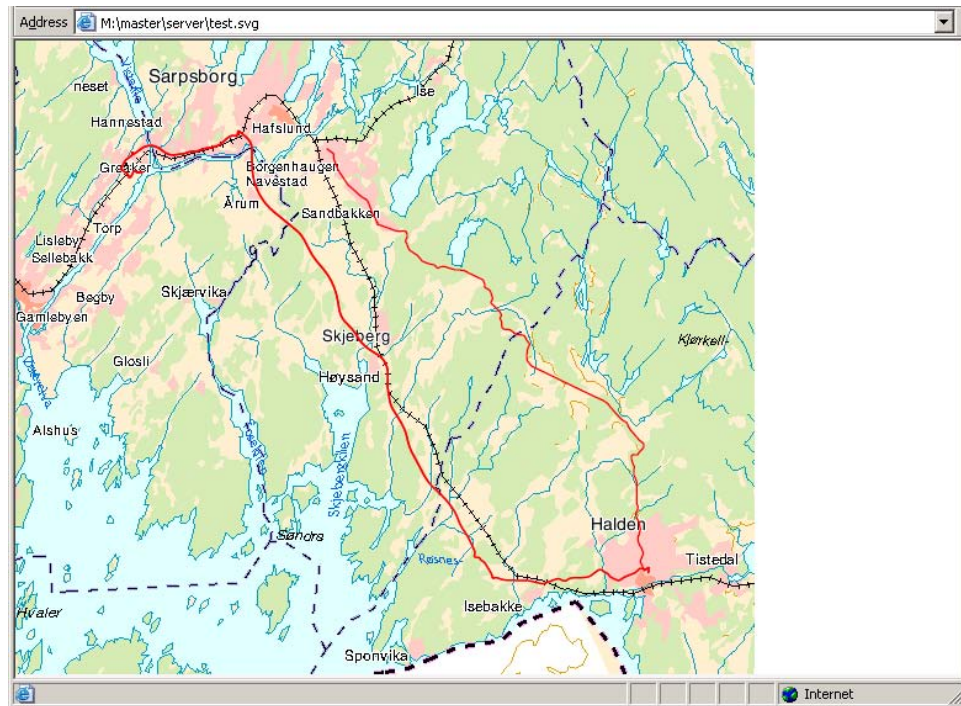



Figure 5.10: SVG Interface: Generated background map from client history

This is then passed to the sleep function in Python's time module:

```
if (real_time_to_wait > 0):  
    time.sleep(real_time_to_wait)
```

The client waits the required amount of milliseconds (giving up processing time to other threads, processes and programs), before continuing to read the NMEA-stream. These time calculations allows the session to have gaps, to be manipulated by 3rd party programs and to be adjusted for a specific situation. Appendix A.1.1 contains a small portion of a log file containing NMEA strings.

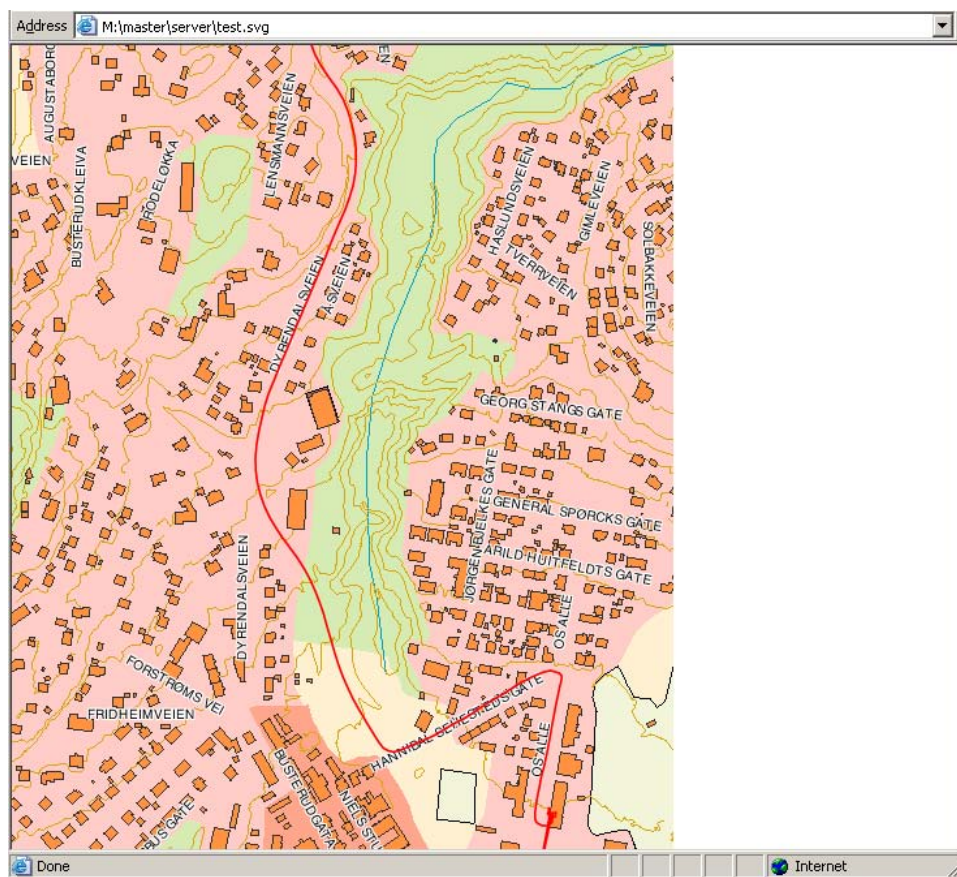


Figure 5.11: SVG Interface: Generated local background map from client history

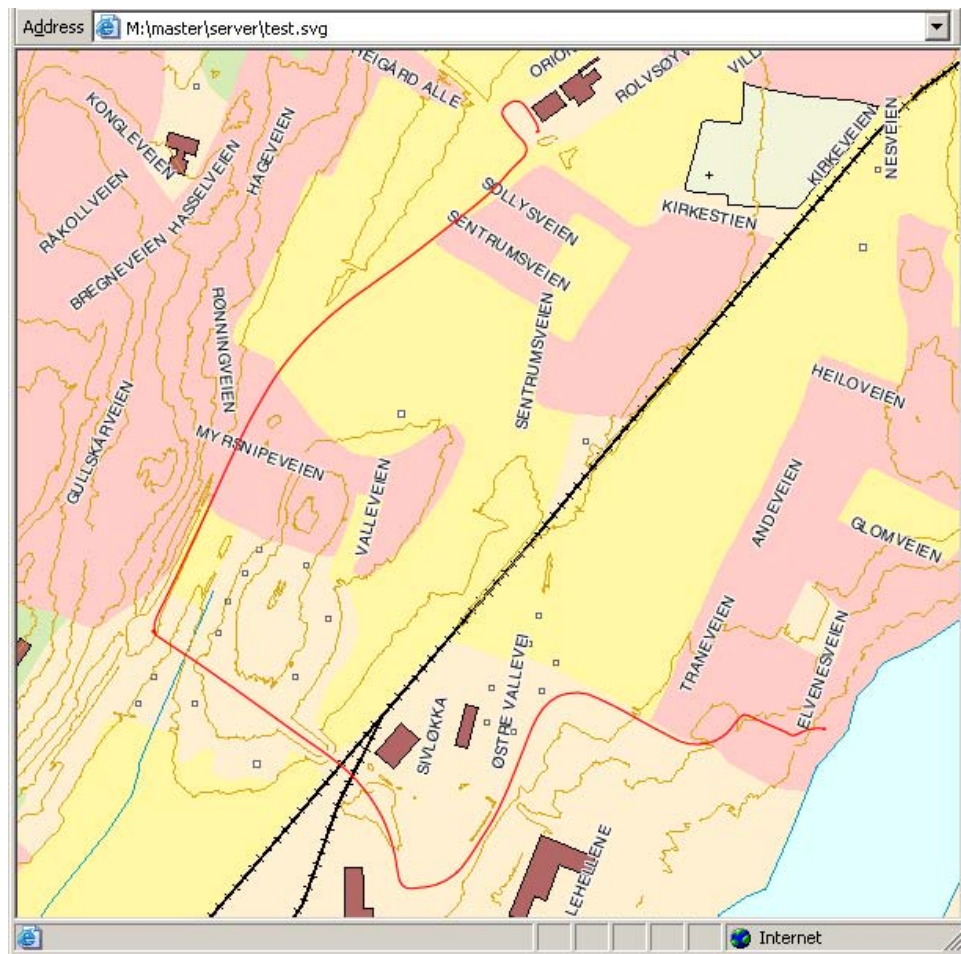


Figure 5.12: SVG Interface: Another generated local background map from client history

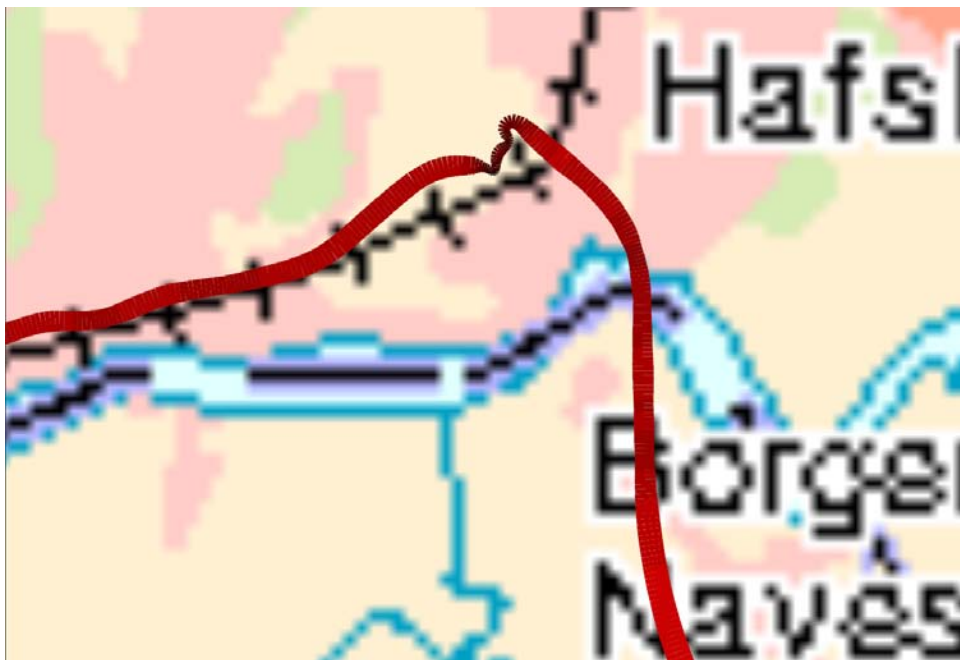


Figure 5.13: SVG Interface: Road segments weighted by velocity

Chapter 6

System Tests

This section contains information about several tests that were run on the prototype implementation. These tests document some of the properties of the system, and provides measurable results regarding the implementation. The first subsection examines one of the scenarios from Section 3.2, while the latter section presents several performance and quality related test results.

6.1 User Case Test

This test case investigates the scenarios described in Sections 3.1 and 3.2. The test was performed in the city center of Fredrikstad during rush hour conditions.

The question posed by the cases is related to the efficiency of choosing an alternate path for reaching your destination. The test area features two distinct roads for arriving at the destination, both requiring the car to pass over the bridge in the city core of Fredrikstad. The primary road is a four lanes wide and is during low hours the best way to get from our point of origin to our destination. The secondary road is a bit older, mostly two lanes wide, but have a few sections which can only fit one car simultaneously. The two different roads are shown in Figure 6.1, where the blue line represents the primary road, and the red line the secondary. Should a driver prefer the secondary road, even if it is slower under ideal conditions?

To establish an overview of the situation, I decided to collect some numbers regarding the main intersection of the road network. Figure 6.2 shows the amount of cars passing through this intersection, divided into five minute intervals. The end and start sections feature only part of the interval, which causes the reported value to be lower in these regions. During interval 19 through 29, the amount of traffic has surpassed the capacity of the surrounding road network. This leads to fewer cars passing through the intersection, and results in delays for drivers in the area. In total, almost 3000 cars passed the intersection each hour. This gives an indication of the quality of information related to the amount of installed clients.

Several tests was performed to determine the traveling time under both optimal and sub-optimal conditions. The results are shown in Table 6.1. The data was extracted from the historic information stored in the Traffic Server, and rendered using SVG. The data was gathered by driving the exact same road segment a multitude of times, and extracting the travel time under different conditions.

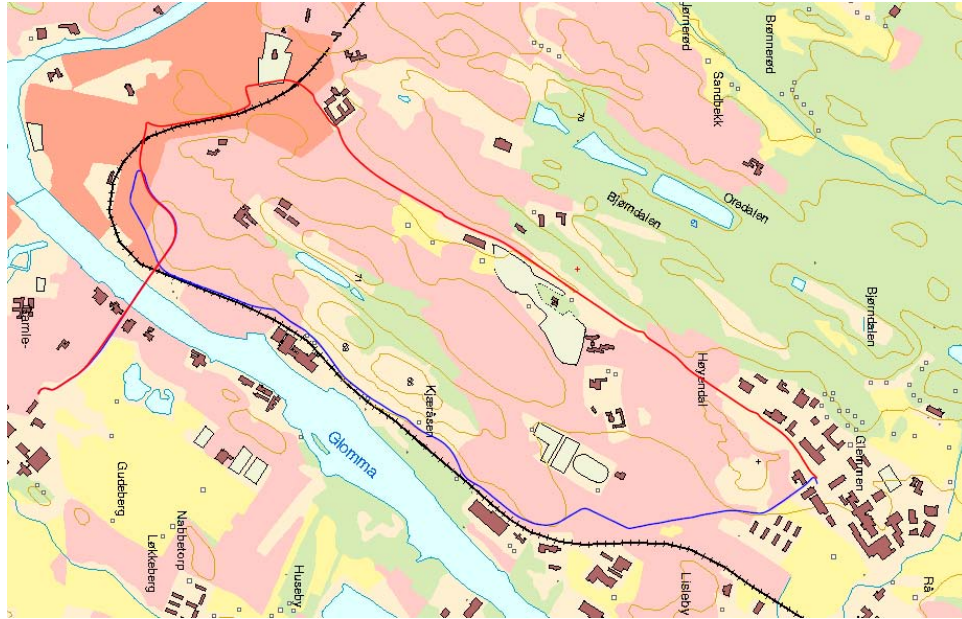


Figure 6.1: The two paths through Fredrikstad city center

Table 6.1: Travel time relative to traffic conditions

Road	Ideal conditions	Regular Conditions	Pre and Post Rush Hour	Rush Hour
Primary	5m 48s	8m 23s	13m 51s	22m 30s
Secondary	8m 18s	8m 49s	11m 58s	15m 47s
Difference	2m 30s	26s	-1m 53s	-6m 43s

My tests show that if a only a small amount of traffic (the amount of installed clients would probably fit this number nicely) is diverted into the old road, the driver will be able to save somewhere around 6 to 7 minutes on what would otherwise be a 22 minute drive. The tests also indicate that traffic warnings should be attached to a audible signal, to indicate that problems has been registered ahead. The number of installed, active clients for maintaining a decent status of the problem area would have to average into one client arriving somewhere along every 15 minutes. With an average of 3000 cars passing through the intersection each hour, an installed base of 0.2% would be sufficient.

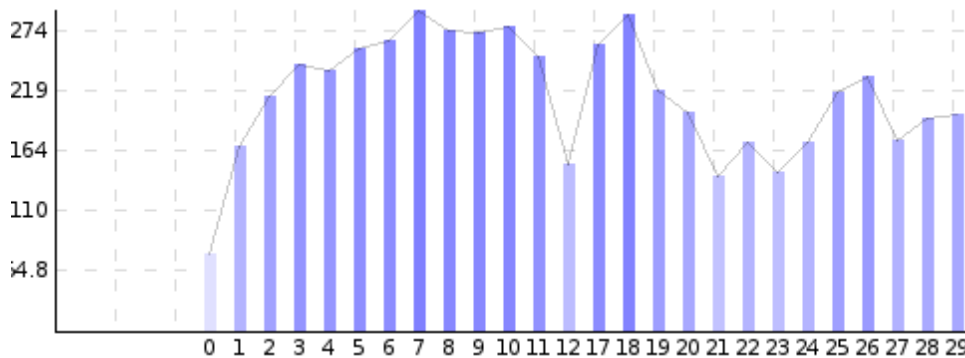


Figure 6.2: Number of cars arriving at intersection divided into five minute intervals

Table 6.2: Average response times from the Traffic Server depending on network connection

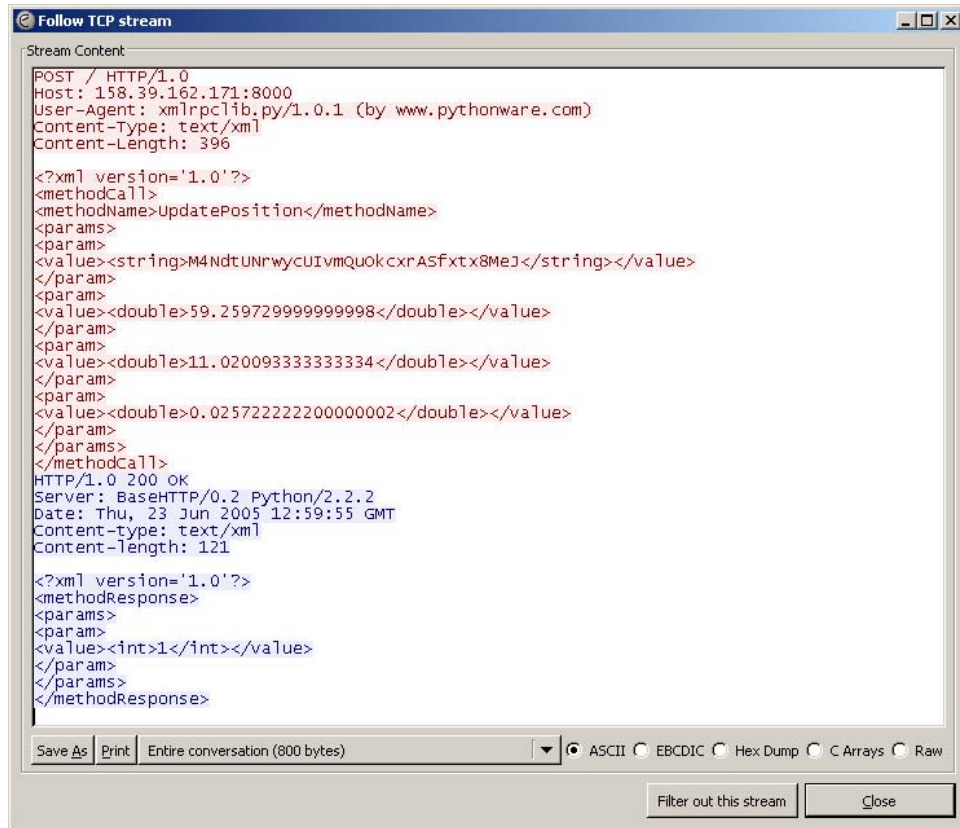
	RegisterEvent	UpdatePosition
UMTS 3G	1.126	5.187
GSM GPRS	2.724	6.805
Emulator	0.380	0.773

6.2 Performance Tests

Table 6.2 features a listing of the response times when connecting to the Traffic Server from different locations. The numbers indicate that the amount of time it took to transfer the information through regular GSM service or 3G is almost identical, while the setup process for a request is more expensive under GSM. I base this conclusion on the indication that the difference between the `RegisterEvent` request (1.598), is close to the observed difference between the `UpdatePosition` request (1.618). Because the `UpdatePosition` request transfers a larger set of data, one would by default expect the UMTS performance to be higher than the GSM performance. This is not the case in the observed transmission times for the dataset. This may be caused by the 3G implementation for data transfer available in Halden at the moment, or may be caused by higher quality data transfer rates being available in the GSM network. This also indicates that report intervals should be at least 10 seconds in size.

The average size of an XML-RPC request and response, the total amount of traffic from a `UpdatePosition` request was 790 bytes. See Figure 6.3 for a complete message, including the HTTP headers.

Figure 6.4 shows the statistics from Ethereal from a session replayed at five times the original speed. Interesting values in this context is the total amount of bytes transferred in the right column, which indicates that a 35 minutes long session would generate about two megabytes of traffic if one were to report the position every second.



```

Stream Content
POST / HTTP/1.0
Host: 158.39.162.171:8000
User-Agent: xmlrpcLib.py/1.0.1 (by www.pythonware.com)
Content-Type: text/xml
Content-Length: 396

<?xml version='1.0'?>
<methodCall>
<methodName>updatePosition</methodName>
<params>
<param>
<value><string>M4NdtUNrwycUivmQuokcxrASfxtx8MeJ</string></value>
</param>
<param>
<value><double>59.259729999999998</double></value>
</param>
<param>
<value><double>11.020093333333334</double></value>
</param>
<param>
<value><double>0.02572222200000002</double></value>
</param>
</params>
</methodCall>
HTTP/1.0 200 OK
Server: BaseHTTP/0.2 Python/2.2.2
Date: Thu, 23 Jun 2005 12:59:55 GMT
Content-type: text/xml
Content-length: 121

<?xml version='1.0'?>
<methodResponse>
<params>
<param>
<value><int>1</int></value>
</param>
</params>
</methodResponse>

```

Save As Print Entire conversation (800 bytes) [Dropdown] [Radio Buttons: ASCII, EBCDIC, Hex Dump, C Arrays, Raw] Filter out this stream Close

Figure 6.3: Transcript of an XML-RPC session from ethereal

Several system tests was performed in regards to determining a delay between client updates. Table 6.3 presents the key figures of actual data transmitted through the network. These figures all uses XML-RPC for data transmission, and can probably be reduced by 90% by using a binary based structure. The "Cost" column has been calculated by using information from the Norwegian network operator Chess [122], and represents actual cost of running the application on your handset. The session used for testing represents 35 minutes of data. The cost of running the client in this configuration drops below one Norwegian krone when the interval is set to 1 minute or higher. If we take into account that we should be able to reduce the amount of transmitted information by a factor of ten, even the interval with a size of 10 seconds becomes plausible for commercial implementation.

The quality difference between the different update intervals is shown in Figure 6.5. The source file (`MergedIntervalTests.svg`) for this application is also available in native SVG format. See Appendix B for more information.

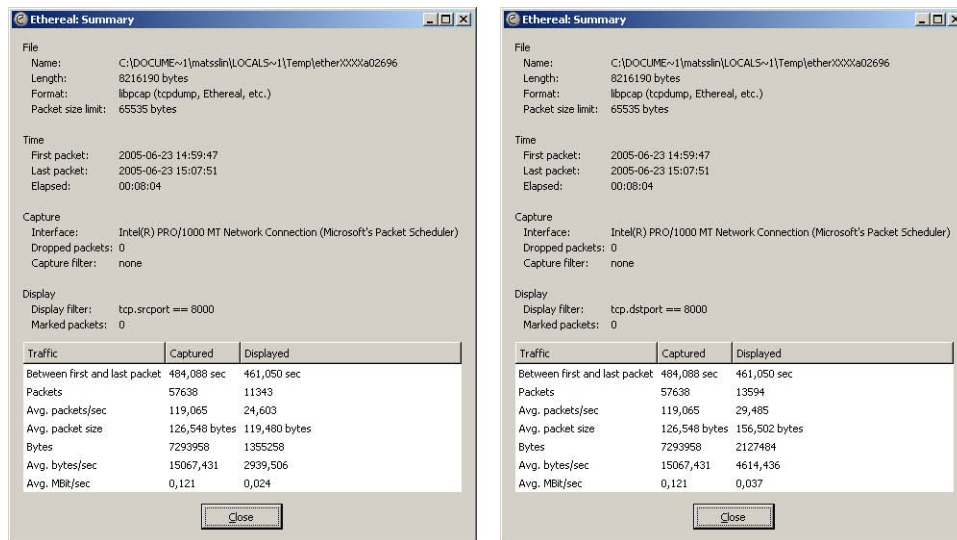


Figure 6.4: Ethereal statistics from a complete 35 minute long session at one second intervals

Table 6.3: Size of transmitted data in relation to update interval

Interval Size	Received Data	Sent Data	Total	Packets	Cost
1 second	2200.81 KB	1224.37 KB	3425.18 KB	2254	47.95 NOK
10 seconds	220.19 KB	123.16 KB	343.35 KB	225	4.8 NOK
30 seconds	73.97 KB	42.70 KB	116.67 KB	75	1.63 NOK
1 minute	37.35 KB	22.45 KB	59.80 KB	38	0.83 NOK
5 minutes	7.90 KB	5.89 KB	13.79 KB	8	0.19 NOK
10 minutes	3.99 KB	3.74 KB	7.73 KB	4	0.11 NOK

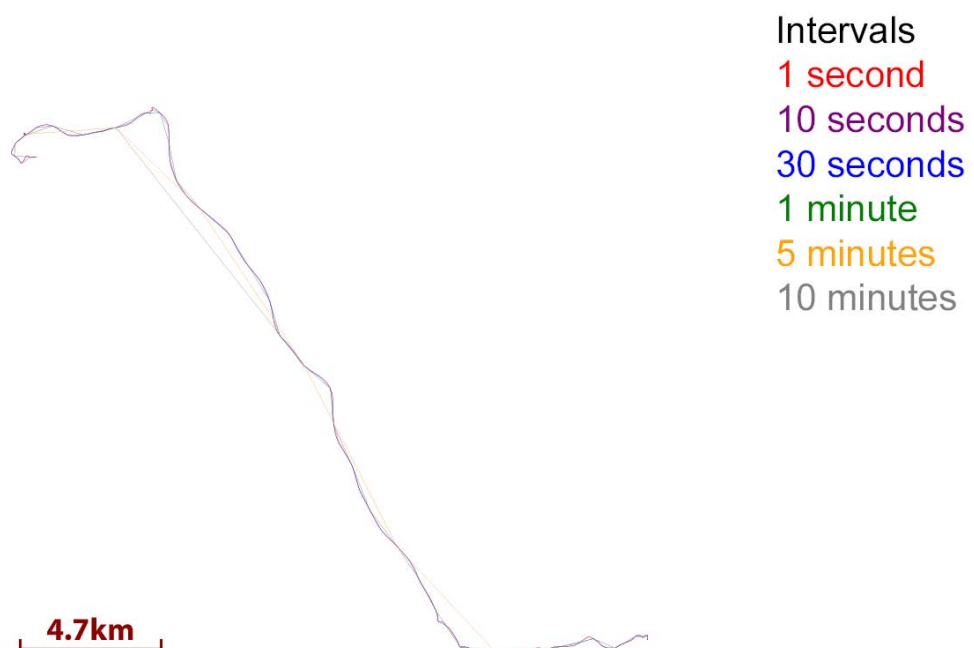


Figure 6.5: Comparison of quality relating to size of update intervals

Chapter 7

Discussion and Future Work

This chapter contains further discussion regarding the results obtained from the implementation and several ideas for further work in the area.

7.1 Discussion

The prototype implementation illustrates the concept described in Chapter 4, and several areas are subject to further discussion based on the experiences from the implementation phase. This section contains a brief discussion of these elements, although further inspection and testing is required to determine their significance. Some of these issues are also mentioned in detail in Section 7.2.

The first area for technical discussion is the usage of XML-RPC for communication between the clients and the Traffic Server. The simplicity and architectural independence of the protocol results in a verbose protocol, requiring more bandwidth to transfer than binary based encoding. The bandwidth issue can be of importance when a monetary cost is attached to every bit and byte that you transfer through the air, and requires some parsing overhead. XML-RPC has been helpful in realizing this project within the time constraint, but the needed bandwidth could probably be reduced by 95% with binary based transmission, because the information transmitted to the server in `UpdatePosition` contains 3 double values (8 bytes on modern architectures). The implementation of XML-RPC on the client and the server applications have proved that the XML-RPC implementation provided with Python is very convenient. The PHP interface provided for XML-RPC is more cumbersome to use, but works as expected after some configuration. The downside to XML-RPC is the missing support for live connections or methods for push communication, and the extensive number of request may prove to be a problem in an extended project. The number of clients supported by a server through XML-RPC may be lower than what one could expect from other communication methods.

The development and application environment used on the mobile devices was Python for Series 60. This proved to be the right choice of platform for a prototype implementation, and allowed me to develop and maintain the applications in a stable and constructive manner. Using PHP for access through HTTP was a good solution, and the integration with SVG and XML-RPC was easily implemented. The first test applications were initially developed using MIDP2, but development

was too cumbersome because of the security limitations. Because of the requirement for non-interactive applications, the Java MIDP2 environment wasn't a suitable platform for this project.

The approach used to determine deviation from the expected value at spatial location is also subject to some discussion. The current implementation uses an evenly spaced grid with 20 meter intervals, and uses the position and heading of the vehicle for lookups in a hash table. This method could be enhanced by better proximity detection, and by mapping the road structure in relevance to existing data in the same area. The grid approach works, and the implementation is simple and straight forward. I suggest that this was the right choice given the constraints of the project, and that the result was as good as one could expect. The algorithm will need adjustments to combat issues regarding natural de-acceleration zones, such as highway junctions and road intersections. This may be solved as the number of observations increase. The method used for the classification of the heading of a vehicle is based on dividing a circle into 16 sections, and choosing the section of the direction of the vehicle. The structure of the border zone between two section can be troublesome with a small number of observations, but will stabilize as the number of observations increase.

7.2 Future Work

There is an extensive potential for more elaborate work within this area of traffic information systems. The community aspect of the information could be enhanced, for example by monitoring vehicles that are regularly close to each other, allow further annotations of the submitted data (such as road quality, media, textual descriptions, road names, etc) and Bluetooth-based detection of persons and vehicles that are in the vicinity of each other. The community aspect of the system can be extended in several directions. The process should be analyzed by conducting user testing, inspection of user behavioral patterns and monitoring the correctness of reported events and their history. This is an interesting and valuable area of testing, and would provide data for future development of the system. The issues with small screens, low memory footprint, hard navigation and other properties of mobile devices should be investigated from the users situation.

The data collection process should investigate the available options and accuracy of network assisted positioning. This would provide a fallback in those situations where an GPS is unavailable and semi-accurate information is necessary. Not much information about these services are available in Norway, and operators should be contacted regarding available services. Such development should attempt to include the OpenLS standard and the core services described therein. The components of an OpenLS-based system would be useful in several other geospatial applications, and could benefit from a community based approach. A geocode/reverse geocode service could make the messages generated in the system more informative for users, with human references to objects.

The automatic detection of potential traffic problems also needs further investigation. The system should have a higher threshold for what it considers a potential problem, and should detect if an area supports both fast and slow moving traffic (such as a toll station, a store close to the road, etc.). The values defining slow traffic, major and minor traffic problems should be inspected according to user preferences. Other ways of determining the expected speed at a location should be evaluated. Road matching could be attempted, but the system is unique because there is not always existing road information at the current location. The topographical extraction of data from the information

should be attempted, and could be related to the available velocity information. It would be interesting to see if it is possible to detect link nodes of road networks by the speed of clients. Intersection tests should also be performed. Manual topographical data collection by mobile devices are another area that could prove interesting, where the user aids the system by manually indicating network connection points.

Storage issues relating to the traffic server should be investigated, and a complete, historical database should be implemented. Not many historical geographical information storage systems are available, in particular not for dynamic data sources. Some introductory thoughts are provided in the implementation specification for the OneMap repository [119], and could be used for further research in the area. An evaluation of how much information it is practical to keep in local memory could also be performed, with reference to the metadata about a road section. The metadata stored about sections could also be investigated, and could be analyzed by traditional data mining techniques.

Regarding the presentation layer several techniques should be investigated further. The current SVG Interface uses a pull method to retrieve continuous information from the server. A possible solution could be to use XML-RPC directly in the SVG document, and other push capable communication methods should be investigated. Other presentation methods such as DHTML [123] and Flash [124] could also be implemented, and an interface conform to the TPEG standard could prove interesting. TPEG would probably generate some interest from commercial vendors, and allow for testing towards existing systems. The distribution of TPEG encoded content to existing clients could also be investigated. An attempt to integrate a geocoded RSS feed into a traditional RSS client could also prove interesting. The possibility of enabling the spatial dimension of RSS files seems interesting, and would be useful for several applications. Other possible formats for exporting data include the standards from the Open Geospatial Consortium. Event and road information could be provided both through WMS and WFS, while grid data could be exported through WCS. The introduction of GML in the data collection layer could also be investigated.

The messages provided by the system should be extended. The current version provides only small amounts of metadata about their context or importance. This should be encoded in the message, together with a classification field. The TPEG standard could be used as a reference for this work, as they already have researched these issues. The messages could also be scrutinized by the community at large, and a personal weight could be introduced for each message type. This would build personal profiles around the message generation framework, and messages could be generated based on the type of clients active. By allowing the community at large to rate the importance of messages, a large set of interesting user information could be extracted from the data set. This would provide an extensive data set for enhancements of the user experience.

Potential security issues should also be investigated. Problem areas include both personal security, such as what information should be kept anonymous, to traditional security, such as disallowing users from retrieving information they do not have access to. The current prototype implementation makes no effort to limit or restrict the information available to any client, other than the requiring the client to log in to the Traffic Server before submitting or editing information.

7.3 In summary

The prototype implementation proved that it is feasible to implement a system like this. Future testing will need to investigate the human element, and determine if people will be involved enough to make the system work. The naive approach to detecting properties that differ from the expected values work to some degree, but must be refined for a future, working system. Road matching may be one possibility. Future versions should also be implemented for multiple platforms, and Java MIDP2 should be investigated further, especially in the context of what warnings can be avoided with signed code. Future implementations could also extend the number of formats that information can be extracted in, and could include other standardized formats like TPEG. Also, exchanging the current transport format of XML-RPC with something else would greatly reduce the cost of running the system.

Chapter 8

Conclusions

The project shows that a community based approach to traffic information has potential, but further testing is required to determine the feasibility of a large scale implementation. In particular, traffic information is well-suited for a community based approach. The information usually stems from the public, and the distribution requirements warrants location based filtering. The frequent changes in the state of the information also accounts for a community based approach. The detection of deviation from the expected values at a location needs further work, but is an interesting part of the system. Several types of traffic events will never be reported if the user manually has to report them, but can nevertheless be important to other users.

The current mobile technology is sufficient for creating the applications needed for such a network, and the capacity of the mobile networks satisfies the bandwidth requirement for clients. The system was developed with fairly low resources, but because of the simplicity of using the XML-RPC and developing for Python for Series 60, performed well. Python for Series 60 simplified the development process of the mobile clients, and made it easy to add new features as needed. XML-RPC is a verbose protocol when used for communication, and is probably unsuitable for a system implemented today. The monetary cost of distributing information by XML-RPC may reduce the participation in the project. By reducing the amount of information transmitted by 90-95%, an hour of updates can be compressed to under the size of an MMS message, and people may be more positive to participating.

The extraction of historic data from the client to create maps worked as expected. The simulations run with 1 second intervals shows that as the number of observations increase, the quality of the extracted road features are more than good enough for regular map quality. The application needs a repository capable of storing historic information between sessions, and because the amount of information will increase with the introduction of new clients, must be effective at retrieval and storage. The Web Map Service provided raster images for the presentation module, and the specification was easy to learn and implement in the Scalable Vector Graphics (SVG) interface. SVG also proved to be a viable strategy, and it will be interesting to see the capabilities of SVG on mobile devices in the future.

The system has proved interesting and viable, and a collaborative approach to traffic system messages looks very promising.

Appendix A

Examples

A.1 NMEA Log Example

A.1.1 Extract from a NMEA log file

```
$GPRMC,095716.862,A,5915.5837,N,01101.2059,E
,0.04,49.98,010605,,,*37
$GPGGA,095717.862,5915.5838,N,01101.2057,E,1,07,1.6,37.4,M,40.5,M
,0.0,0000*4D
$GPGSA,A,3,11,20,01,24,07,14,25,,,,,,,,,3.3,1.6,2.9*3E
$GPRMC,095717.862,A,5915.5838,N,01101.2057,E
,0.03,31.24,010605,,,*38
$GPGGA,095718.862,5915.5838,N,01101.2056,E,1,07,1.6,37.5,M,40.5,M
,0.0,0000*42
$GPGSA,A,3,11,20,01,24,07,14,25,,,,,,,,,3.3,1.6,2.9*3E
$GPGSV
,3,1,10,20,66,240,41,01,53,079,43,11,52,160,37,24,46,287,32*79
$GPGSV
,3,2,10,07,36,289,40,25,24,097,34,14,22,045,35,23,18,194,00*72
$GPGSV,3,3,10,05,07,005,00,04,05,299,00*7C
$GPRMC,095718.862,A,5915.5838,N,01101.2056,E
,0.05,51.18,010605,,,*39
$GPGGA,095719.862,5915.5839,N,01101.2055,E,1,07,1.6,37.7,M,40.5,M
,0.0,0000*43
$GPGSA,A,3,11,20,01,24,07,14,25,,,,,,,,,3.3,1.6,2.9*3E
$GPRMC,095719.862,A,5915.5839,N,01101.2055,E
,0.07,16.33,010605,,,*32
$GPGGA,095720.862,5915.5839,N,01101.2054,E,1,07,1.6,37.9,M,40.5,M
,0.0,0000*46
$GPGSA,A,3,11,20,01,24,07,14,25,,,,,,,,,3.3,1.6,2.9*3E
$GPRMC,095720.862,A,5915.5839,N,01101.2054,E,0.04,88.31,010605,,,*3
F
$GPGGA,095721.862,5915.5839,N,01101.2053,E,1,07,1.6,38.0,M,40.5,M
,0.0,0000*46
```

```
$GPGSA,A,3,11,20,01,24,07,14,25,,,,,,,,,3.3,1.6,2.9*3E
$GPRMC,095721.862,A,5915.5839,N,01101.2053,E,0.05,28.68,010605,,*3E
$GPGGA,095722.862,5915.5840,N,01101.2052,E,1,07,1.6,38.0,M,40.5,M,0.0,0000*4A
$GPGSA,A,3,11,20,01,24,07,14,25,,,,,,,,,3.3,1.6,2.9*3E
$GPRMC,095722.862,A,5915.5840,N,01101.2052,E,0.05,42.73,010605,,*34
```

A.2 RSS Feed Examples

A.2.1 Georeferenced RSS Feed Showing The Three Categories of Traffic Slowdowns

```
<rss version="2.0" xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel>
    <title>Traffic Server</title>
    <link>http://www.ia-stud.hiof.no/~matsslin/traffic/index.php?rss=all</link>
    <description>An interactive, georeferenced traffic information service built after the principles of collaboration and community exchange of information. Supports ?rss=&lt; bounding box&gt; for extraction of information for a given area. Use ?rss=all to retrieve all registered events, regardless of location.</description>
    <language>en-us</language>

    <item>
      <title>Major traffic slowdown</title>
      <link>http://www.ia-stud.hiof.no/~matsslin/traffic/index.php?event_id=Mlzm230M&format=html</link>
      <description>Major traffic slowdown</description>
      <dc:Author>31337</dc:Author>
      <dc:Date>Mon, 20 Jun 2005 13:28:31 +0200</dc:Date>
      <dc:Identifier>Mlzm230M</dc:Identifier>
      <dc:Covrage>11.012905,59.266343333333</dc:Covrage>
      <dc:Description>Major traffic slowdown</dc:Description>
      <dc:Source>Reported</dc:Source>
      <dc:Rights>Public Domain</dc:Rights>
    </item>

    <item>
      <title>Minor traffic slowdown</title>
      <link>http://www.ia-stud.hiof.no/~matsslin/traffic/index.php?event_id=qciXWgNw&format=html</link>
      <description>Minor traffic slowdown</description>
      <dc:Author>31337</dc:Author>
      <dc:Date>Mon, 20 Jun 2005 13:28:06 +0200</dc:Date>
```



```

    <dc:Identifier>qciXWgNw</dc:Identifier>
    <dc:Coverage>11.02009,59.259731666667</dc:Coverage>
    <dc:Description>Minor traffic slowdown</dc:Description>
    <dc:Source>Reported</dc:Source>
    <dc:Rights>Public Domain</dc:Rights>
  </item>

  <item>
    <title>Traffic Jam</title>
    <link>http://www.ia-stud.hiof.no/~matsslin/traffic/index.
      php?event_id=A4bfy9sl&format=html</link>
    <description>Traffic Jam</description>
    <dc:Author>31337</dc:Author>
    <dc>Date>Mon, 20 Jun 2005 13:28:23 +0200</dc>Date>
    <dc:Identifier>A4bfy9sl</dc:Identifier>
    <dc:Coverage>11.012943333333,59.266333333333</dc:Coverage>
    <dc:Description>Traffic Jam</dc:Description>
    <dc:Source>Reported</dc:Source>
    <dc:Rights>Public Domain</dc:Rights>
  </item>
</channel>
</rss>

```

A.3 SVG Basic Examples

A.3.1 Red Circle

```

<?xml version="1.0"?>
<svg width='200' height='200' viewBox='0 0 200 200'>
  <circle cx='100' cy='100' r='50' style='fill: red' />
</svg>

```

A.4 XML-RPC Request and Response Example

From [2].

A.4.1 Request

```

<?xml version="1.0"?>
<methodCall>
  <methodName>examples.getStateName</methodName>
  <params>
    <param>
      <value><i4>41</i4></value>
    </param>
  </params>
</methodCall>

```

```
        </param>
      </params>
    </methodCall>
```

A.4.2 Response

```
<?xml version="1.0"?>
<methodResponse>
  <params>
    <param>
      <value><string>South Dakota</string></value>
    </param>
  </params>
</methodResponse>
```

Appendix B

External Documentation

This chapter contains references to external documentation that was submitted as part of this master thesis.

B.1 Traffic Server

Documentation for the Traffic Server class and its methods are available on the accompanying CD under `prototype/documentation/`.

Bibliography

- [1] Open Geospatial Consortium Inc. *OpenGIS Location Services (OpenLS): Core Services*, May 2005.
- [2] Dave Winer. XML-RPC. URL: <http://www.xmlrpc.com/spec>.
- [3] Lars Wischhof, André Ebner, and Hermann Rohling. SOTIS A Self-Organizing Traffic Information System. Institute of Electrical and Electronics Engineers (IEEE), 2003.
- [4] Route 66 Mobile Europe 2006. URL: <http://www.66.com/route66/products.php?cid=NO\&sec=0\&ssec=1\&prodid=1576>.
- [5] Wayfinder Navigator. URL: http://www.wayfinder.com/products/product_en.php?id=30.
- [6] Berkman Center for Internet & Society. RSS 2.0 Specification. URL: <http://blogs.law.harvard.edu/tech/rss>.
- [7] Bo Leuf and Ward Cunningham. The wiki way: Collaboration and sharing on the internet.
- [8] Barry H. Kantowitz, Richard J. Hnaowski, and Susan C. Kantowitz. *Driver Reliability Requirements for Traffic Advisory Information*, pages 1–22. 1997.
- [9] Tom Clements. Making Sense of Cellular, July 2002. URL: <http://developers.sun.com/techttopics/mobility/getstart/articles/radio/>.
- [10] Wikipedia. Nordic Mobile Telephone. As of 29. June, 2005, URL: http://en.wikipedia.org/wiki/Nordic_Mobile_Telephone.
- [11] UMTS Forum. URL: <http://www.ums-forum.org/>.
- [12] John Scourias. A Brief Overview of GSM. URL: <http://kbs.cs.tu-berlin.de/~jutta/gsm/js-intro.html>, 1994.
- [13] GSM Association. History of GSM. URL: <http://www.gsmworld.com/about/history/index.shtml>, 1997.
- [14] Eli Biham and Orr Dunkelman. Cryptanalysis of the a5/1 gsm stream chiper. *Lecture Notes In Computer Science*, 1977:43–51, 2000.

- [15] Simon Buckingham. What is General Packet Radio Service? URL: <http://www.gsmworld.com/technology/gprs/intro.shtml>, January 2000.
- [16] Telenor ASA. URL: <http://www.telenor.no/>.
- [17] NetCom A/S. URL: <http://www.netcom.no/>.
- [18] NRK Norsk rikskringkasting. URL: <http://www.nrk.no/>.
- [19] TV2. URL: <http://www.tv2.no/>.
- [20] The Series 60 Community. Series 60 platform. URL: <http://www.series60.com/>.
- [21] Python Software Foundation. Python programming language. URL: <http://www.python.org/doc/Summary.html>.
- [22] James Kardach. Bluetooth architecture overview. *Intel Technology Journal*, 5(2), 2000.
- [23] Christer Edvartsen. OneMap IO Client for Palm OS. URL: <http://starcrow.starzinger.net/~cogo/digmap.pdf>.
- [24] Open Mobile Alliance. WAP Forum. URL: <http://www.openmobilealliance.org/tech/affiliates/wap/wapindex.html>.
- [25] Gunnar Misund and Mats Lindh. Annotating Mobile Multimedia Messages with Spatiotemporal Information.
- [26] Java Community Process. *JSR 75: PDA Optional Packages for the J2ME Platform*, June 2004. URL: <http://www.jcp.org/en/jsr/detail?id=75>.
- [27] Qusay Mahmoud. Getting started with the FileConnection APIs, December 2004. URL: <http://developers.sun.com/techtopics/mobility/apis/articles/fileconnection/>.
- [28] Java Community Process. *JSR 135: J2ME Mobile Media API (MMAPI)*, June 2003. URL: <http://java.sun.com/products/mmapi/>.
- [29] Symbian Ltd. *SYMBIAN OS SDK V8.1A, Symbian OS Guide, Multimedia*, 2005. URL: http://www.symbian.com/developer/techlib/v8.1adocs/doc_source/guide/Multimedia-subsystem-guide/index.html#doc_source%2eMultimedia%2esub%2eguide%2egen%2eindex.
- [30] Java Community Process. *JSR 82: Java APIs for Bluetooth*, March 2002. URL: <http://www.jcp.org/en/jsr/detail?id=82>.
- [31] C. Enrique Ortiz. Using the Java APIs for Bluetooth Wireless Technology, Part 1 - API Overview, December 2004. URL: <http://developers.sun.com/techtopics/mobility/apis/articles/bluetoothintro/>.

- [32] Open Geospatial Consortium. Open Geospatial Consortium. URL: <http://www.opengeospatial.org/>.
- [33] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, and François Yergeau. *Extensible Markup Language (XML) 1.0 (Third Edition)*, February 2004. URL: <http://www.w3.org/TR/2004/REC-xml-20040204/>.
- [34] Simon Cox, Paul Daisey, Ron Lake, Clemens Portele, and Arliss Whiteside. *OpenGIS® Geography Markup Language 3.0(GML) Implementation Specification*. URL: https://portal.opengeospatial.org/files/?artifact_id=7174.
- [35] Dublin Core. Dublin Core XML Schemas, <http://dublincore.org/schemas/xmls/>. URL: <http://dublincore.org/schemas/xmls/>.
- [36] Don Murray and Juan Chu Chow. OGC Standards in Practice, 2004. URL: http://www.gmldays.com/gml2004/papers/OGC_Standards-DonMurray.pdf.
- [37] World Wide Web Consortium. The Semantic Web. URL: <http://www.w3.org/2001/sw/>.
- [38] Open Geospatial Consortium Inc. *OpenGIS Web Map Server Cookbook*, April 2003. URL: <http://www.ogcnetwork.org/docs/03-050.pdf>.
- [39] Statens Kartverk. The Arealis Project. URL: <http://www.statkart.no/IPS/?template=arealis>.
- [40] Bob Raskin. Guide to Distributing Your Data Products Via WMS 1.1.1 - A Tutorial for Data Providers. Technical Guide.
- [41] Demis, DEcision support and Management Information Systems. URL: <http://www.demis.nl/>.
- [42] ESRI, Environmental Systems Research Institute. URL: <http://www.esri.com/>.
- [43] ESRI. ArcIMS. URL: <http://www.esri.com/software/arcgis/arcims/index.html>.
- [44] The GeoServer Project. GeoServer. URL: <http://geoserver.sourceforge.net/html/index.php>.
- [45] MapServer. URL: <http://mapserver.gis.umn.edu/>.
- [46] GPX: the GPS Exchange Format. URL: <http://www.topografix.com/gpx.asp>.
- [47] Groundspeak Inc. Geocaching - The Official Global GPS Cache Hunt Site, 2000–2005. URL: <http://www.geocaching.com/>.
- [48] Lil Devil. GPX Spinner. URL: <http://www.gpxspinner.com/>.

- [49] GpxView. URL: <http://strandberg.org/gpxview/>.
- [50] CacheMate. URL: <http://www.smittyware.com/palm/cachemate/>.
- [51] magnalox - magnificent GPS logs and interactive reports:. URL: <http://www.magnalox.net/>.
- [52] TrailRegistry - Interactive Maps and Custom Trail Guides for Hiking and Backpacking. URL: <http://trailregistry.com/trailregistry/index.jsp>.
- [53] World Wide Web Consortium. POIX: Point Of Interest eXchange Language. URL: <http://www.w3.org/TR/poix/>.
- [54] The World Wide Web Consortium. URL: <http://www.w3c.org/>.
- [55] URL: <http://www.nmea.org/>. The national marine electronics association.
- [56] Johan Koolwaaij. The GSM Cells Project. URL: <http://www.lab.telin.nl/~koolwaaij/showcase/gsmcells/>.
- [57] GSM Association. Location Based Services. Permanent Reference Document: SE.23, URL: <http://www.gsmworld.com/documents/lbs/se23.pdf>, January 2003.
- [58] Ajay Magon and Reena Shukla. LBS, the ingredients and the alternatives. URL: <http://www.gisdevelopment.net/technology/lbs/techlbs006.htm>.
- [59] Mobile in a Minute. Mobile Positioning. URL: http://www.mobilein.com/mobile_positioning.htm.
- [60] Peter H. Dana. Global Positioning System Overview. Informal Article, URL: http://www.colorado.edu/geography/gcraft/notes/gps/gps_f.html, September 1994.
- [61] Martin Selmayr and Mona Lund. Cars that can dial 112: Commission and industry target 2009. Press release, URL: http://europa.eu.int/information_society/activities/esafety/text_en.htm, February 2005.
- [62] U.S. Coast Guard. Coast Guard Maritime Differential GPS Service. URL: <http://www.navcen.uscg.gov/dgps/default.htm>.
- [63] O. Mezentsev, Y. Lu, G. Lachapelle, and R. Klukas. Vehicular Navigation in Urban Canyons Using a High Sensitivity GPS Receiver Augmented with a Low Cost Rate Gyro.
- [64] Rebecca Blood. *Weblogs: A History and Perspective*, pages 7–16. Jun 2002.
- [65] Tom Nicolai, Nils Behrens, and Heidi Thielemann. 'be a freeporter!': Enabling a mobile news publishing community. URL: http://matrixpn.auriga.wearlab.de/freeporter-mgain_presentation/nicolai04beafreeporter.pdf.

-
- [66] Joan Walsh. Who killed Dan Rather? URL: <http://www.salon.com/opinion/feature/2005/03/09/rather/>, Mar 2005.
- [67] Wikipedia. URL: <http://www.wikipedia.org/>.
- [68] CNN. Top new york times editors quit. URL: <http://www2.cnn.com/2003/US/Northeast/06/05/nytimes.resigns/>, Mar 2004.
- [69] Dan Gillmor. *We the Media*. O'Reilly Media, 1 edition, Jul 2004.
- [70] Wikinews. URL: http://en.wikinews.org/wiki/Main_Page.
- [71] Verdens Gang (VG). Verdens Gang (VG). URL: <http://www.vg.no/>.
- [72] Online Computer Library Center. Dewey Decimal Classification. URL: <http://www.oclc.org/dewey/>.
- [73] W3C. Resource Description Framework (RDF). URL: <http://www.w3.org/RDF/>.
- [74] W3C. OWL Web Ontology Language Overview. URL: <http://www.w3.org/TR/2004/REC-owl-features-20040210/>.
- [75] M. Nilsson. *ID3 tag version 2.4.0 - Main Structure*, Nov 2000. URL: <http://www.id3.org/id3v2.4.0-structure.txt>.
- [76] Christopher L. T. Brown. Exchangeable Image File Format. Technical White Paper, URL: <http://www.techpathways.com/uploads/ExIF.pdf>, Oct 2004.
- [77] Marc Davis. Mobile Media Metadata: Metadata Creation System for Mobile Images.
- [78] Marc Davis and Risto Sarvas. Mobile Media Metadata for Mobile Images.
- [79] Marc Davis, Nancy van House, Jeffrey Towle, Simon King, Shane Ahern, Carrie Burgener, Dan Perkel, Megan Finn, Vijay Viswanathan, and Matthew Rothenberg. MMM2: Mobile Media Metadata for Mobile Sharing.
- [80] Kris Cardinaels, Michael Meire, and Erik Duval. Automatic metadata generation: the simple indexing interface. International World Wide Web Conference, May 2005.
- [81] Erik Duval and Wayne Hodgins. Making Metadata go away: "Hiding everything but the benefits". Dublin Core Conference, 2004.
- [82] Charlotte Jenkins, Mike Jackson, Peter Burden, and Jon Wallis. Automatic RDF Metadata Generation for Resource Discovery.
- [83] RDS Forum. URL: <http://www.rds.org.uk/>.
- [84] Blaupunkt. URL: <http://www.blaupunkt.com/>.

- [85] Michael A. Mollenhauer, Melissa C. Hulse, Thomas A. Dingus, Steven K. Jahns, and Cher Carney. *Design Decision Aids and Human Factors Guidelines for ATIS Displays*, pages 23–61. 1997.
- [86] Robert Graham and Val A. Mitchell. *An Evaluation of the Ability of Drivers to Assimilate and Retain In-Vehicle Traffic Messages*, pages 185–201. 1997.
- [87] Jill Flemming, Paul Green, and Stewart Katz. Driver Performance and Memory for Traffic Messages: Effects of the Number of Messages, Audio Quality, and Relevance. Technical Report, Jun 1998.
- [88] Ana Lúcia C. Bazzan, Joachim Wahle, and Franziska Klügl. *Agents in Traffic Modelling - from Reactive to Social Behaviour*, pages 303–307. 1999.
- [89] Elisabet Thompson. Integrating PDA, GPS and GIS technologies for Mobile Traffic Data Acquisition and Traffic Data Analysis. 2003.
- [90] The World DAB Forum. URL: <http://www.worlddab.org/>.
- [91] European Broadcasting Union (EBU). URL: <http://www.ebu.ch/>.
- [92] TPEG Forum. URL: <http://www.tpeg.org/>.
- [93] British Broadcasting Corporation. URL: <http://www.bbc.co.uk/>.
- [94] Helen Eveleigh. DATEX and DATEX Profiles. Briefing Note, URL: <http://www.centrico.ten-t.com/documents/briefing%20notes/DATEX.pdf>.
- [95] Yasuhiko Kajiya, Yuuji Yamagiwa, Yasuhiro Kudo, Hidekazu Kagaya, and Takafumi Shimano. Road Web Markup Language - XML for Road Information Distribution on the Net. URL: <http://www2.ceri.go.jp/sirwec2002/english/papers/kajiya.pdf>, 2002.
- [96] Society of Automotive Engineers (SAE). URL: <http://www.sae.org/servlets/index>.
- [97] P4. URL: <http://www.p4.no/>.
- [98] Kanal 24. URL: <http://www.kanal24.no/>.
- [99] Statens Vegvesen. Nasjonal Vegdatabank (NVDB). URL: http://www.vegvesen.no/1042188916633/vegvesen-Page-SVVsubSideInnholdMal_1069341123047.html.
- [100] Microsoft Corporation. URL: <http://www.microsoft.com/>.
- [101] Microsoft AutoRoute. URL: <http://www.microsoft.com/uk/homepc/autoroute/default.msp>.

-
- [102] Microsoft MapPoint. URL: <http://www.microsoft.com/mappoint/default.msp>.
- [103] Microsoft Streets and Trips. URL: <http://www.microsoft.com/streets/default.msp>.
- [104] MSN Maps and Directions. URL: <http://mappoint.msn.com/>.
- [105] Route 66. URL: <http://www.66.com/>.
- [106] TomTom. URL: <http://www.tomtom.com/>.
- [107] Appello. URL: <http://www.appello.se/>.
- [108] Telia Sonera. URL: <http://www.teliasonera.se/>.
- [109] Digi.no. Svenskene får veiviser på mobilen. URL: <http://www.digi.no/php/art.php?id=215140>.
- [110] World Wide Web Consortium. *Scalable Vector Graphics (SVG) 1.1 Specification*, 2003. URL: <http://www.w3.org/TR/SVG/>.
- [111] Microsoft. Visio. URL: <http://office.microsoft.com/en-us/FX010857981033.aspx>.
- [112] Adobe. Illustrator. URL: <http://www.adobe.com/products/illustrator/main.html>.
- [113] Adobe. SVG Zone. URL: <http://www.adobe.com/svg/>.
- [114] ECMA. ECMAScript Language Specification. URL: <http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf>.
- [115] RSS-DEV Working Group. Rdf site summary (rss) 1.0. URL: <http://web.resource.org/rss/1.0/>.
- [116] Wikipedia. Multi media card, June 2005. As of 22. June 2005, URL: http://en.wikipedia.org/wiki/Multi_Media_Card.
- [117] The WDDX Project. WDDX: The Web Distributed Data eXchange. URL: <http://www.openwddx.org/>.
- [118] Intel. Intel IXP4XX Product Line of Network Processors and IXC1100 Control Plane Processor: Understanding Big Endian and Little Endian Modes, December 2003. Application Note, URL: <http://www.intel.com/design/network/applnotes/25423701.pdf>.
- [119] Kristian Lunde, Mats Lindh, and Bjørn Håkon Horpestad. The OneMap Repository. URL: <http://www.onemap.org/~mats/OneMapRepository.pdf>, 2005.

-
- [120] TinyLine. URL: <http://www.tinyline.com/>.
- [121] PHP Hypertext Preprocessor. URL: <http://www.php.net/>.
- [122] Chess Communication AS. URL: <http://www.chess.no/>.
- [123] Evan Golub and Ben Shneiderman. Dynamic query visualizations on World Wide Web clients: a DHTML solution for maps and scattergrams. *International Journal of Web Engineering Technology*, 1(1):63–78, 2003.
- [124] Macromedia Flash MX 2004. URL: <http://www.macromedia.com/software/flash/?promoid=BINT>.