Tallinn Technical University

Interoperability of Geographical Information Systems

by

Hanno Kuus

THESIS

submitted in partial fulfillment of the requirements for the degree of
Master of Science in Engineering

Tallinn 2002

Declaration

I hereby declare that the current thesis ‘Interoperability of Geographical Information Systems’ is my own original work, submitted to the Tallinn Technical University in partial fulfillment of the requirements of the degree of Master of Science, and it has not been submitted for reception of an academic degree before.

Hanno Kuus

Tallinn, May 2002

Interoperability of Geographical Information Systems

Hanno Kuus

Abstract

There are many Geographical Information System (GIS) applications in use today, unfortunately we are also used to various problems regarding the spatial data. An important aspect of every system is interoperability—the ability of GIS software to use the spatial data, provided by other systems. It is also important not to overlook the quality of the data, i.e. whether the accuracy, structure and content of the exchanged set of data conform to our requirements.

The purpose of this thesis is to give an overview of two prominent methods for providing interoperability in Geographical Information Systems—data files and geospatial data services. At first the reader is got acquainted with basic concepts of Geographical Information Systems. In the next chapter the properties of several file formats are described and compared for fitness for practical tasks.

Data files have been the traditional method for exchanging data among applications for a long time but different services are getting more and more attention as spatial data sources nowadays. This thesis focuses mainly to web services, developed by the Open GIS Consortium (OGC). The *Web Map Service (WMS)* and *Web Feature Server (WFS)* service standards are getting closer examination. A comparative view of two covered interoperability methods (files and services) is also provided.

The following chapters concentrate to the implementations of the OGC Web Map Service. Three different WMS servers are compared in regard of building various spatial data exchange services. A detailed look to the architecture and functioning of one server application is also provided. Another chapter is dedicated to WMS client applications, where four programs are compared and analyzed for suitability for practical situations.

The thesis contains also descriptions of three projects that are using the Open GIS Consortium Web Map Service specification for solving practical problems in the field of GIS interoperability.

Andmevahetus geoinfosüsteemides

Hanno Kuus

Kokkuvõte

Geoinfosüsteemid on tänapäeval laialt levinud, kahjuks oleme harjunud ka mitmesuguste probleemidega ruumiandmetega töötamisel. Üheks oluliseks küsimuseks ruumiandmete kasutamisel on andmevahetus—tuleb veenduda, et kasutatav tarkvara suudab kasutada sisendandmeid teistest süsteemidest, pöörates seejuures tähelepanu ka andmete kvaliteedile: kas andmete täpsus, sisu ja struktuur on sellised, nagu me ootame.

Käesoleva magistritöö eesmärgiks on anda ülevaade kahest geoinfosüsteemides kasutatavast andmevahetuse viisist—andmefailidest ja andmevahetusteenustest. Töö algab geoinfosüsteemide põhimõistete tutvustamisega, seejärel võrreldakse mõningate laialt levinud failivormingute omadusi ning nende kasutatavust praktilistes ülesannetes.

Traditsioonlise failivahetuse kõrval on tänapäeval olulisteks ruumiandmete allikateks saamas ka andmevahetusteenused. Antud töös on tähelepanu pööratud peamiselt OpenGIS konsortsiumi poolt loodud veebipõhiste teenuste tutvustamisele. Lähemalt on kirjeldatud kaardiserveri (*Web Map Service (WMS)*) ja objektiserveri (*Web Feature Server (WFS)*) teenuseid. Toodud on ka kahe käsitletud andmevahetusmeetodi (failide ja teenuste) omavaheline võrdlus.

Magistritöö teises pooles on lähema vaatluse alla võetud kaardiserveri teenuse rakenduslik külg. Võrreldud on kolme WMS serveri poolt pakutavaid võimalusi andmevahetusteenuse realiseerimiseks, detailsemalt on tutvustatud ühe serverprogrammi arhitektuuri ning funktsioneerimist. Eraldi peatükk on pühendatud ka kaardiserveri klientprogrammidele, kus on võrreldud nelja erineval tehnoloogial põhinevat rakendust ning analüüsitud nende sobivust praktikas aset leidvatesse situatsioonidesse.

Magistritöö sisaldab ka kolme rakendusprojekti kirjeldust, kus on kasutatud OpenGIS konsortsiumi kaardiserveri standardit reaalses elus esilekerkivate probleemide lahendamiseks.

Acknowledgments

No man is an island and this work could have not been produced without the assistance of many people. First I would like to thank my supervisor, professor Vello Kukk, for his encouragement and guidance throughout my studies. I am also thankful to professor Ennu Rüstern, head of the Department of Computer Control, for his constant encouragement. My appreciation goes to the whole academic staff of the Tallinn Technical University and the Institute of Geography of the Tartu University for educating and directing me.

Special thanks belong to Peep Krusberg for numerous discussions and suggestions, and to Peeter Väling for his help in finalizing the thesis. I would also like to thank my colleagues in the *R-Süsteemid* company for many ideas and discussions, and for creating a pleasant working atmosphere.

I am deeply grateful for my wife and son for their constant support and persuasion, and for providing necessary relaxation.

Finally I should mention my thankfulness and respect to the numerous researchers who have developed the huge body of knowledge that was available to me in printed form and electronically in the Internet.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Today, computers are around us everywhere, performing tasks varying from simple to extremely complex. Geography is among the fields of research where computers are of great assistance nowadays, especially since the advent of the *Geographical Information System (GIS)* at the end of the 1950s, that revolutionized map-making, affected significantly the data analysis and visualization techniques, and, furthermore, opened many new opportunities to make use of geospatial data. These days, in the era of personal computers, GIS is used in many areas by a very wide range of specialists in their everyday work in solving multifarious problems.

The commercial GIS industry, having its roots in the 1970s, is still growing at a steady pace, about 20 per cent, each year [36]. This has led us to the situation where there are numerous great software packages from various firms for building our Geographical Information Systems. However, the standardization process in this area started only recently. The lack of universally adopted standards has produced a situation, where the communication between different systems is made hard and sometimes impossible. There have been efforts for creating standards but these activities have mostly remained local to some nationality or group of interest. In the 1990s, at last, several international standardization projects took off. These major players, like Open GIS Consortium (OGC), International Organization for Standardization (ISO) Technical Committee (TC) 211, Digital Geographic Information Working Group (DGIWG) and others, have crafted several well-balanced standards for regulating the GIS community. All these organizations are still busy in improving their specifications and creating new standards, making this world a better place for all GIS users.

The author of this paper is working in the *R-Süsteemid*[1] company, located in Tallinn, Estonia. I have had a chance of working on several projects that used geographical data in this small software engineering and consulting office. These projects have demonstrated for me that the importance of open standards cannot be underestimated and the best way for building systems is to adhere to public specifications and promote them where applicable.

The topic of this thesis, the methods of communication (*interoperability*) of Geographical Information Systems, is very important in the field of geomatics, and more important, it affects the life of most practical GIS users. Even more, the enhanced interoperability that is emerging from the new standards may increase the usage of geospatial data, bringing new people to the world of GIS by enabling them easily supplement their databases with locational data. The importance of spatial information and Geographical Information Systems are constantly increasing, sometimes they are even viewed as the keys to the future of democracy [49].

[1] http://www.r-systems.ee/

1.1 Goals

The purpose of this thesis is to give an overview of the field of interoperability of Geographical Information Systems, especially the development of the latest standards. I would also like to demonstrate some practical aspects of applying these standards, how they change the life of both GIS software developers and end users.

The main goals of my work are:

- Analyzing the problems of interoperability in GIS, appropriate standards and practices. Two methods—files and services are covered.
- Introducing to the wider audience some newly available technologies in the field of communications of Geographical Information Systems.
- Showing some ways for practically using open standards for increasing the interoperability and availability of GIS services. For this purpose special software products are created and combined with existing programs. These undertakings are among the first projects in Estonia that make use of OpenGIS web services.

1.2 Composition

The thesis is composed of seven chapters, here I outline of the main points from each of them, starting from chapter 2, where a short introduction to GIS world is made with an emphasis to the problems that are discussed in the following chapters. Additional topics include a short presentation of metadata and standardization of geodata.

In the 3$^{\text{th}}$ chapter two methods of communication and interoperability in the world of GIS are analyzed. The first part of the chapter covers file exchange, several formats that are currently used for exchanging geographical data are described and compared. The rest of chapter 3 concentrates to the web-based communication methods. An overview of such methods is provided, as well as more detailed description of two OGC specifications, Web Map Service (WMS) and Web Feature Server (WFS). The chapter ends with conclusions drawn from the presented material.

The rest of the thesis is devoted to the WMS protocol. Chapter 4 on page 35 presents several samples of WMS server software. The main focus will be on the architecture of F-Open server (a product of the *R-Süsteemid* company, carrying a substantial contributions by the author of this thesis) but a couple of alternatives are also discussed. The chapter concludes with comparison of those map servers.

The next chapter (Chapter 5 on page 46) is devoted to WMS client programs. Several clients for different computing platforms are described and compared.

Chapter 6 looks into some possibilities of using the WMS standard for solving problems that arose from practical necessities. Three projects are described and the implications of engaging the WMS standard are highlighted.

Final conclusions are drawn and the key points are reiterated in the 7$^{\text{th}}$ chapter. Directions for theoretical and practical work for the future are provided in that chapter as well.

The expected audience of chapters 4 and 5 are developers of GIS systems, while the content of chapters 3 and 6 should be interesting to both developers and end users alike.

Chapter 2

Concepts of GIS

Geographical Information Systems are widely used in every field of our everyday life, often we use them without even noticing it. But still the concepts of GIS are not very well understood, even by computer literate people. The purpose of this chapter is to make the reader, who may possess a lot of knowledge about information systems, acquainted with spatial data processing and spatial information systems[1].

A *Geographical Information System (GIS)* is an information system ('A mechanism used for acquiring, filing, storing, and retrieving an organized body of knowledge [4].') which contains some spatial information, that is data for phenomena on, above or below the Earth's surface. Spatial data can be in many different forms: satellite images of the Earth, census reports in tabular form or digital vector map of city streets [30]. A Geographical Information System also *uses* the spatial component of the data, either for visualization or for spatial analysis. This property differentiates GIS from other systems, e.g. the information system of a library may also contain maps (in paper and/or digital forms), but as long as it only stores the data and does not use it, it is not a Geographical Information System.

It has to be noted that nobody has been successful to provide an universal definition of GIS. The reason is quite simple—similar to other complex phenomena, the GIS has many variations and is used for many different purposes, giving people disparate impressions about the essence of a GIS (what parts and operations are important, what components does it contain, etc.). Some possible definitions together with the discussion on the topic are provided in the article by D. J. Maguire ([32]).

The outline of a sample GIS is shown in Figure 2.1 on the following page. From this figure one can see the complexity of a system—operating such systems definitely needs the knowledge from many domains. Real systems differ from this example, either in hardware, e.g. they are lacking a digitizer or have an added speech synthesizer, or in software, for example instead of a Database Management System (DBMS) they use operating system's file service. In this paper I will cover only the software side of GIS and will concentrate to the data storage and interfacing questions as well as to some important properties of the data (both spatial and attribute data).

[1]Some authors prefer to use the term *spatial* instead of geographical, as the concepts and properties also apply to other fields where co-ordinates on multiple axes (not necessarily related to the Earth) are used (e.g. component placing on electrical circuit boards). In this paper I use these words as synonyms because I will focus on Earth-related spatial data.

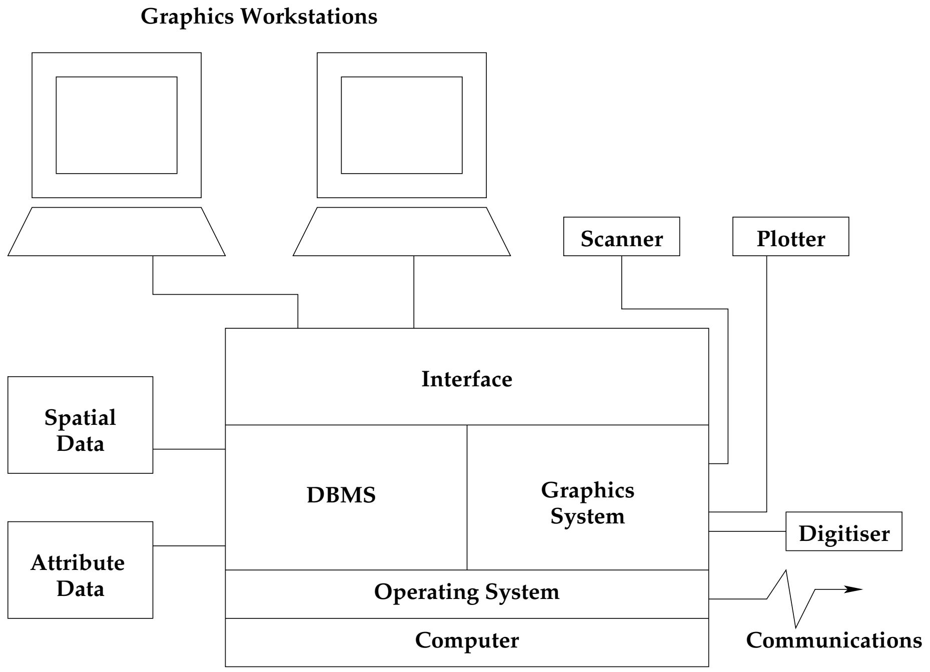


Figure 2.1: The physical components of a GIS [2].

2.1 GIS Data Models

The single most important component of every GIS is its data. A GIS usually stores two different kinds of data: spatial data and attribute data (aspatial data). The difference between them lies in interpretation, the physical representation can be absolutely the same. Attribute data is often stored in a DBMS and treated in the same way as in other information systems (used for making queries and reports) while spatial data is used for visualization or spatial operations—spatial queries, spatial analysis and other operations. Note that these two kinds of data are not separated (at least from the viewpoint of a user) but usually tightly integrated. Typically systems have capabilities for querying attribute data, based on location (spatial filtering) or *vice versa*, selecting only the pieces from spatial data that have some specific attribute values.

Spatial data can be presented in many forms but all digital representations[2] can be divided into two fundamental classes—raster or vector data.

This division originates from our everyday cognition of the space around us (the naïve geography)—sometimes we look to the objects that fill the space and have spatial attributes, other times we like to pinpoint a location in the space and enumerate the objects (or more precisely, the attributes of space) that we found there. These cognitive methods are used interchangeably, depending on the properties of the objects (discrete, continuous, relative size, etc.), our experience and other factors. The first approach can be formalized as a vector model and the second as a raster model [11, 15].

Raster model (raster data) is defined as a tessellation of a plane into regular cells (*pixels*)[3], each one representing a value of some quantity in given location. The whole dataset represents the field of given quantity varying across the space, spatial position of each cell is implicit in the ordering of the pixels. The value of the cells of certain areas may be irrelevant to our goal but in the raster dataset the pixels are present (and having some values)

[2]Here I do not consider narrative forms for storing spatial data as they are not practical in most cases.
[3]Raster data is therefore sometimes called *grid-cell* data.

Property	Raster model	Vector model
Data entry	Faster	Slower
Data structures	Simple	Complex
Positional accuracy	Limited by the size of pixels	Unlimited
Attributes	Good for continuous attribute fields	Good for intrinsic attributes of map features
Data analysis	Good for spatial analysis and modelling	Good for spatial inquiries
Storage required	Larger	Smaller
Transformations of co-ordinates	Problematic	Easy

Table 2.1: Comparison of raster and vector models [27].

nevertheless. The cells are usually square-shaped, but in principle they can be any uniform shape that completely covers the plane without gaps, such as triangles, hexagons or some fractal shapes [30, 45]. Raster data plays important rôle in remote sensing applications and other systems where the entities are continuous in space, like a temperature field.

Vector model, on the other hand, contains individual *objects* (*features*) which usually represent real-world phenomena. The spatial components of each object are composed of points, lines, areas (polygons) and volumes. Only the co-ordinates of special points (end-points of lines, corners of polygons, etc.) are stored in the dataset, making it more compact than raster data—no co-ordinate information is kept for a location where no objects are found. The spatial part of vector data can be treated as a *graph* with all properties and operations that are defined for graphs also applying for vector data. Objects have usually attributes. An *attribute* is a chunk of character data associated with given object (each attribute value is similar to the value of a pixel in raster model). An object can possess many attributes, giving the user the ability to perform various queries on the data [2].

One important property that can be present in a vector dataset is topology. *Topology* is generally defined as the spatial relationships between adjacent features, usually in a two-dimensional space. In GIS, topology is implemented through the structure of data. Topological data structures are advantageous because they provide an automated way for handling digitizing and editing errors and artifacts, and reduce the storage that is needed for polygons because boundaries between neighbouring polygons are stored only once. Spatial relationships such as adjacency, connectivity and containment, as well as spatial analysis methods are much easier to implement on topological data structures [54]. Sometimes topology moves from secondary quality of a dataset to the focus, there exist tasks (like the mapping of principal schema of bus lines of a city) for which the topological relations are more important than co-ordinates and topological space is used instead of Euclidean space for plotting such maps (i.e. only topology is shared between the map and the reality, not the properties of metric spaces (distance)) [18].

Topology has different implications for raster data, as the uniform picture elements of a raster image cannot support the same set of topological relations that we know for vector model (or from everyday life). However, a *hybrid raster* model is proposed, obtained by completing the raster cell with edges and nodes on its boundaries into a hybrid cell complex. Such hybrid raster systems support traditional topological relations in Euclidean space and can be used as a basis for hybrid Geographical Information Systems where (from the user's point of view) the data is not divided between vector and raster models [60].

Level	Abstraction	Description
0	Reality	The real world phenomena as they exist.
1	Conceptual model	Components and relationships, pertaining to the specific phenomena thought to be relevant. This data model is independent of specific systems or data structures.
2	Logical model	The logical organization of the data and the manner in which relationships between components are defined.
3	Physical model	The physical layout of the data in computer's memory (file structure).

Table 2.2: Levels of data abstraction in GIS [45].

The choice of the data model should be based on the problem to be solved, the intrinsic properties of the data, the capabilities of the computer soft- and hardware, and the experience of the personnel. Some comparison points between the vector and raster models, shown in Table 2.1 on the page before should help to make the right choice. The focus in this thesis will be on vector data.

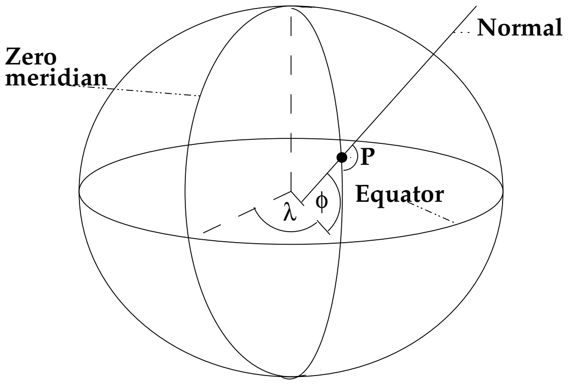
The main purpose of every Geographical Information System is to model the reality. The real world is incredibly complex and is not well suited for data processing. Digital spatial data always represents a certain model of the real world. The *conceptual data model* provides necessary simplifications and regularity for storing and processing the data. The data model contains nearly always several levels, each level refining and regulating the previous one. Various abstraction schemas have been proposed for spatial data models, Table 2.2 describes one such schema.

2.2 Co-ordinates

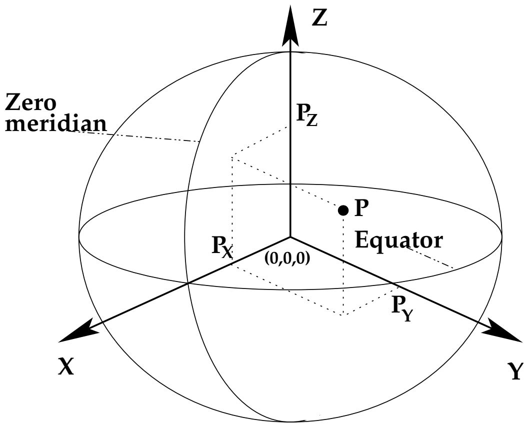
The location of spatial data is expressed using the *co-ordinates*. The *co-ordinate system* defines the rules for transforming the real-world (Earth) co-ordinates into map co-ordinates and *vice versa*. The values of co-ordinates of a point are meaningful only when we know the datum that was used for determining the co-ordinate values. A *geodetic datum* defines the reference system that describes the size and shape of the Earth. There are many co-ordinate systems in use for different purposes, most are modelling the shape of the Earth (the *geoid*) as an ellipsoid (a sphere is good enough approximation for small scale mapping and planar model can be used for very large scale surveying). Many ellipsoidal models of the Earth have been defined during the history, most of which are meant for using only in particular region. The advent of satellite navigation systems has increased the employment of geocentric datums (which are usable for all points on the Earth), most notably the World Geodetic System (WGS) 84 [51].

There are many types of co-ordinates used for mapping purposes. Choosing the right system may be challenging but it will result in smaller size, better handling, and simpler processing of the data [34]. The three most commonly used types are shown in Figure 2.2 on the next page.

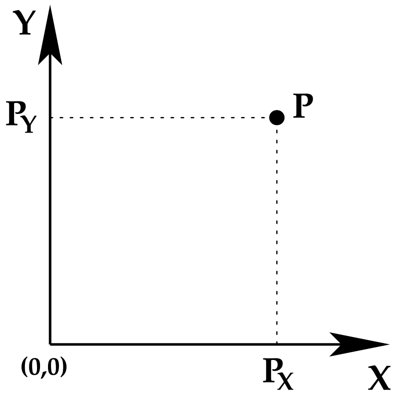
The *geodetic co-ordinates* (geographic co-ordinates) (λ, φ) of a point (Figure 2.2, a) are defined by the ellipsoidal co-ordinates of the projection of the point onto the surface of a reference ellipsoid along the normal of that ellipsoid. Geodetic *latitude* φ is defined as the inclination of the normal to the equatorial plane of the ellipsoid. The *longitude* λ is the



(a) Geodetic $P = (\lambda, \varphi)$.



(b) Geocentric Cartesian $P = (X, Y, Z)$.



(c) Cartographic $P = (X, Y)$.

Figure 2.2: Co-ordinate systems.

angle between the meridional plane (a plane through the normal and the minor axis of the reference ellipsoid) of the point and the zero meridian [39].

The three-dimensional geocentric *Cartesian co-ordinates* (Figure 2.2, b) with the origin at the centre of the reference ellipsoid are sometimes used (A common need for such system is during the transformation of co-ordinates from one datum to another). The Z-axis is taken perpendicular to the equator, X-axis is directed along the equatorial plane to the zero meridian. The Y-axis is perpendicular to the other two. With Earth-centered co-ordinates we can use the traditional mathematical apparatus for Cartesian co-ordinates and need not worry about the angular entities of the ellipsoidal co-ordinates. The downside of using this system is the inconvenience of computing the co-ordinates—changing the position or height of the observer involves computation of new values for all three co-ordinates (the values of geodetic co-ordinates do not change when the height is varied). Nevertheless, all satellite-based navigation systems use the three-dimensional Cartesian co-ordinates internally.

The *cartographic co-ordinates* (also planar co-ordinates, grid co-ordinates, Figure 2.2, c) are Cartesian co-ordinates on a plane that are obtained by transforming the co-ordinates from the three-dimensional space onto the plane using a *map projection* (analytical) formulae or numerical methods (polynomial transformations). Two-dimensional co-ordinates are much better displayed and managed on planar media (such as paper or computer screen). The big problem with this approach is the impossibility of distortion-less representation of the Earth's surface (therefore the distance, measured with ruler on a planar map would be incorrect in most cases). There exist map projections which preserve some properties of the ellipsoidal Earth, for example area, local angles, scale, or direction. Comprehensive explanations of many map projections together with appropriate formulae are provided by J. P. Snyder ([50]).

The *L-EST* co-ordinate system which is used by the Estonian Basic Map and most other spatial databases in Estonia is based on the Lambert Conformal Conic projection with two standard parallels ($58°00'$ and $59°20'$), European reference frame EUREF-89 and the GRS-80 ellipsoid [3].

2.3 Metadata

One very important aspect of GIS, sometimes unfortunately underrated in importance, is metadata. *Metadata* (metainformation) is generally defined as data about data, although in some contexts or applications it may have slightly different (usually more specific) meaning. The supplementary and descriptive data is distinguished from the actual data (the data being described) only by its content. Metadata provides an architecture, describing a dataset (the raw data) within a data environment [44].

Spatial datasets are generally large and their structure tends to be complex (especially when using vector data model) and quite different from one database to another, therefore the importance of metadata is even greater for geographical data, comparing to completely aspatial databases.

Metadata can be divided into several categories ([44]):

Access metadata The logical structure of the data and other information needed for queries and retrieval. Definitions of object classes, attributes, and attribute values. Also the information about co-ordinate system and relevant parameters.

Semantic metadata The purpose of the data, feature classification guidelines, decision rules regarding the representation of spatial and aspatial data, spatial integrity constraints, etc.

Quality metadata The qualitative parameters that characterize the dataset. For spatial databases it includes the accuracy (spatial and temporal) of data, time of collection (or source if compiled from previous work) and other parameters.

Transfer metadata The explanation of the data transfer between applications.

Storage metadata Description of the storage management system (how and where the data is stored, how often archived, etc.). These parameters are very useful for systems which contain huge amounts of data on various storage media.

There may be additional categories for providing additional explanations of the data.

Metadata (especially the parts which define the structure of the database, feature classes, attributes, etc.) is often called *data dictionary* [30]. Several spatial data exchange standards (e.g. S-57 and DIGEST) define data dictionaries that represent the conceptual models (describing the abstractions of the real world) for the respective applications.

The rapidly increasing amount of data that is being produced has led us to the point where using the metadata is essential for gaining an overview of the available data and for being able to select interesting pieces from the huge pile of raw data.

Some spatial data storage and interchange formats (see Chapter 3.1) provide means for storing metainformation together with raw data. Such formats provide usually a predefined set of fields for storing metadata which may be incapable of storing e.g. additional organization-wide data. Also some datasets may be stored in forms that do not provide such fields. Therefore an organization-wide metadata database is often needed to keep track of the various datasets and their qualities. One such system is described in a paper by Michael J. Meitner *et al.* ([37]). Such databases should store metainformation in a strict, machine-processable format (there should not be possible to enter 'TM' into the projection field for one dataset and 'Transverse Mercator' for another, provided that the projection is the same for both). Additional, free-form comments should also be encouraged. When the metainfo database cannot be applied then the descriptive files (preferably in plain text format) should accompany each spatial dataset.

The *Dublin Core Metadata Initiative*[4] is an open forum engaged in development and promotion of interoperable online metadata standards and practices. The *Dublin Core Metadata Element Set* (`http://dublincore.org/documents/dces/`) describes various metadata elements for common data, such as the title, subject, date, but also type, source and relations, total of 15 elements. Dublin Core is promoting metadata for all kinds of databases, not only for spatial data. Dublin Core Metadata Elements contain no special information for spatial content.

Content Standard for Digital Geospatial Metadata (CSDGM)[5] is a specification specialized to the digital geospatial metadata, created by the Federal Geographic Data Committee of the United States of America (USA). The objectives of this recommendation are to provide a common terminology and definitions for documentation of digital geospatial data. The standard includes descriptions for various elements in several categories: identification, data quality, data organization, spatial references, entities (features) and attributes, data distribution, citations, temporal characteristics, metadata reference and contacts [13]. The extensive set of metadata elements defined by this standard should provide adequate for most GIS projects.

The ISO/TC 211 is developing a standard 15046-15 (*Geographic information—Metadata*) for specifying a common schema for metainformation for spatial datasets.

2.3.1 Importance of Metadata: An Example

Co-ordinates (either in 2D or 3D space) do not describe spatial data absolutely. Without knowing the origin, directions, and units of the co-ordinate axes[6] the actual values of co-ordinates are meaningless.

When dealing with geographical data in cartographic co-ordinates we also need to take into account different projections and datums, as the co-ordinate values for any given feature depend on the method of mapping the geographical co-ordinates into planar ones.

Figure 2.3 shows two maps of the same area but using different co-ordinate systems, layered on top of each other. When features have to be copied from one of those maps (database) to another their locations would be incorrect, unless co-ordinates are transformed to the datum and projection of the destination map. The appropriate transformations can only be performed when there is precise knowledge about the source and destination co-ordinate systems.

[4] http://dublincore.org/

[5] http://www.fgdc.gov/metadata/contstan.html

[6] Assuming Cartesian co-ordinates. Similar requirements can be made for other types of co-ordinate systems.

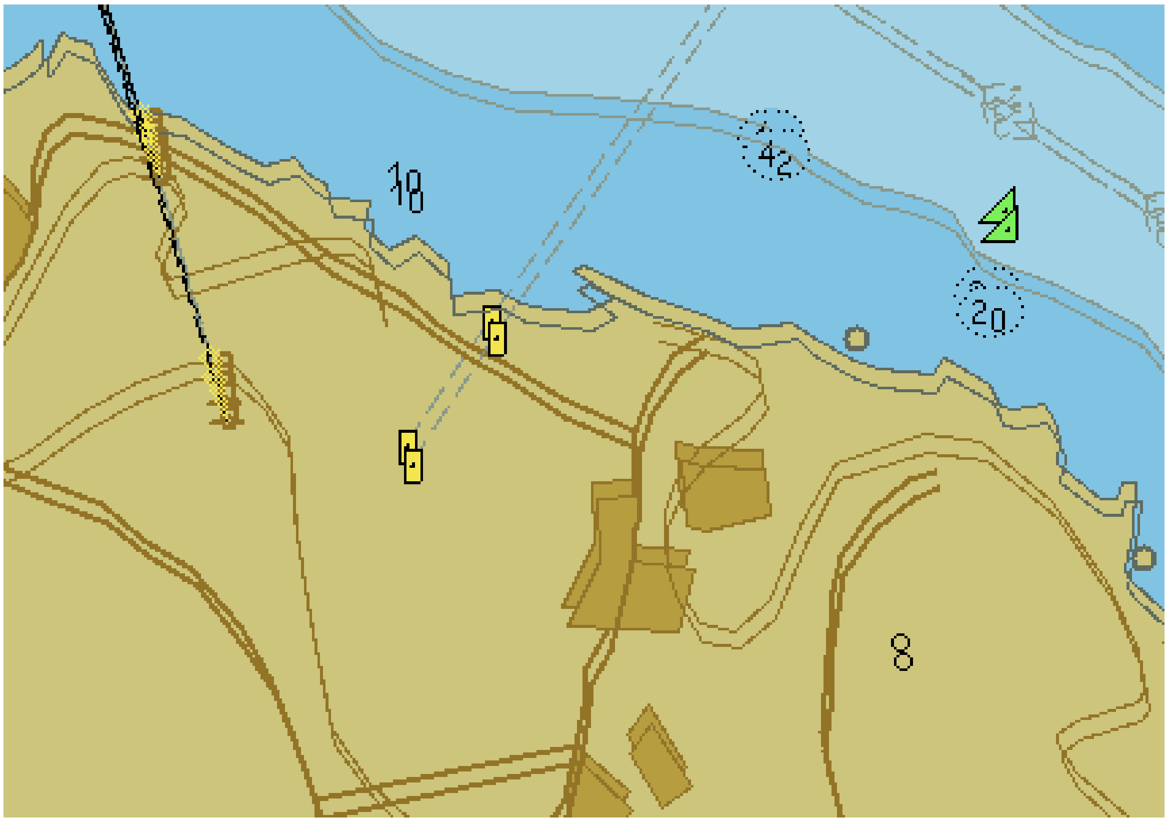


Figure 2.3: Two maps of the same area with different co-ordinate systems set on top of each other. The lower map is in the *L-EST* co-ordinate system (Lambert Conformal Conic projection), upper map uses the *TM-Baltic* co-ordinate system (Transverse Mercator projection). Scale 1:22000, co-ordinate shift is about 50 m.
Northern coast of Muhu island, Estonian Nautical Chart sheet no. 513.

The importance of metainformation being accompanied with any dataset is clearly perceptible from this example, as we cannot interpret the values of co-ordinates otherwise. The interoperability of different systems (data sources) is also virtually impossible without valid formal description (metadata) about the data sources.

2.4 Standardization

The standardization of spatial data and other aspects of GIS has taken off only in recent years, following the years of *de facto* standards, set by large corporations. The active movement for standardization and interoperability (which is virtually impossible without formal standardization) during the last decade has begun to produce new, high quality standards that are created by international organizations using the principle of consensus. Those standards are usually recommendatory but following them is still the only way out of the quagmire of incompatible systems.

International Organization for Standardization (ISO)[7] is the biggest international standard body whose members are national standard organizations. ISO has approved over 13000 different standards for all domains. International Organization for Standardiza-

[7] http://www.iso.ch/

tion has set up a Technical Committee (TC) 211 (*Geographic information/Geomatics*)[8] for standardizing the handling of digital geographical data. This committee has affiliations to many other standard-making organizations (International Hydrographic Organization (IHO), OGC, DGIWG and others) for establishing uniformity and interoperability of standards.

ISO/TC 211 has a plan to produce over thirty standards, most of which are still unreleased drafts. The topics of those standards range from the spatial model to the multimodal location based services and cover the whole range of problems in the handling and processing geospatial data. Selected topics include data quality, terminology, spatial and temporal schemas, different spatial referencing systems, qualification of personnel, and web map server.

There are also other ISO committees dealing with spatial data, e.g. the Geographic Data File (GDF) standard, created by the ISO TC/204 (*Transport information and control systems*).

The Open GIS Consortium (OGC)[9] is an international organization, founded in 1994, aimed at developing open standards for geoprocessing technologies and growing interoperability in Geographical Information Systems. The members of the OGC are companies, universities and research organizations around the world that participate in a consensus process.

OGC provides an *abstract specification* which contains a framework and reference model for implementation specifications. It is a high-level guide, defining the theoretical data models, principles, etc. *Implementation specifications* are papers that clarify and specify in detail the interfaces of some part of the abstract specifications. The implementation specifications are written for software developers. Open GIS Consortium may also provide *recommendation papers* which present consortium's stand on several topics. Recommendation papers are usually produced as preliminary versions of standards and may develop into an implementation specification or into a part of the OpenGIS abstract specification.

The implementation specifications include several standards that define interfaces for providing ubiquitous processing of geographical data in different environments. The examples of OGC services include map server, feature server, geoparser, gazetteer, and others. OGC services are examined in greater detail in section 3.2 on page 25. All OGC standards (both approved and candidate specifications) are publicly available from the consortium's web page.

The process of creating and approving an OGC standard is significantly faster process than approving ISO standards. The fast development cycle, especially at the beginning, and the open nature of the standardization process help the software vendors to implement the proposed specifications and providing valuable feedback to the standards committee. The end users also benefit from the open process, as they can use the specifications and software implementations of the preliminary versions of the standard to gradually adopt and refine a particular application of the standard.

ISO and OGC are not the only organizations that have undertaken the efforts of specifying and standardizing some aspects of geospatial data processing. The other players usually operate on national level (e.g. the Spatial Data Transfer Standard (SDTS), developed by US Geological Survey (USGS)[10]) or, in case of international organizations, they co-ordinate the activities in a strictly specialized field (International Hydrographic Organization (IHO)[11]

[8] http://www.isotc211.org/
[9] http://www.opengis.org/
[10] http://www.usgs.gov/
[11] http://www.iho.shom.fr/

the creator of S-57 (IHO Transfer Standard for Digital Hydrographic Data) and many other standards for hydrography, is a good example of such organizations).

Smaller projects build also on top of the shoulders of ‘giants’ (OGC, ISO, etc.), such as the GIS Interoperability Project Stimulating the Industry in Europe (GIPSIE) project, stimulating the interoperability in European GIS industry and seeking answers to some questions that are not very important to the OpenGIS Consortium (whose members are mostly from the USA) but are essential in Europe, for example multi-language support and interoperability in national level [61].

Chapter 3

Interoperability

The previous chapter showed the great complexity and diversity in the field of Geographical Information Systems. The variety of different systems has risen the question of connectivity and interoperability between systems. This chapter gives an overview of the needs and forms of communications in the GIS field. Then a more detailed analysis is done upon some published standards with the emphasis on file exchange formats and web-based services.

The term *interoperability* is used to denote co-operation or communication. The standard *ISO 19104 Geographic Information—Terminology* defines the interoperability as:

> 'The capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units [19].'

The hardware of modern computers is quite unified and poses few problems—almost all computers use 8-bit bytes, have the ability of reproduce graphics, etc., the software layers that reside 'on top' of hardware (operating systems, compatibility libraries) provide additional levelling. Perhaps the most notable problem that arises from the hardware of computers is the *endianness*—different ordering of bytes in memory words. The machines with *big-endian* processors (e.g. Sun Scalable Processor Architecture (SPARC)) have trouble reading binary files, written on computers with *little-endian* processors (like Intel Pentium)[1]. The trouble only affects binary files, text files can be transformed from one to another without any problems. Of course, the variety of peripheral devices is much greater, sometimes leading to problems when a device cannot be used with some particular computer's hardware.

In general, one should not have many interoperability problems with today's hardware. This is not exactly the situation in GIS software and from now on the focus in this chapter will be implicitly on software of Geographical Information Systems.

Although GIS has been along for several tens of years the problems of interoperability in GIS software had not been regulated by international standards up to now. Data was usually exchanged in proprietary file formats, some of which gained the status of *de facto* standard (for example *shapefiles* from ESRI or design files (*dgn*) from Intergraph/Microstation), regardless on the fact whether the specifications were publicly available or not. In larger projects seamless work-flow was assured only by careful choosing (or writing new) of software components, perhaps from only one vendor. Some projects did so good work that parts of their systems became *de facto* standards, e.g. the Topologically Integrated Geographic

[1] The problem is easily solvable in software, one needs only know whether the file record is written in big-endian byte order (memory words are stored most significant byte first) or little-endian byte order (memory words are stored least significant byte first) and swap bytes when necessary.

Encoding And Referencing System (TIGER) project of the United States' Census Bureau [55].

In this work two aspects of interoperability are covered. First, an overview of existing GIS file formats is provided, then there are outlined several services providing spatial data in one form or another.

While the file exchange will undoubtedly remain very important part of data accessibility processes in Geographical Information Systems, *services* provide many new possibilities for data transfer. Services in this context are protocols, which make possible data sharing between the *server* and *clients*. Such services usually build one or more layers upon existing accessibility protocols[2]—Common Object Resource Broker Architecture (CORBA), Component Object Model (COM), Hypertext Transfer Protocol (HTTP), etc. Files are usually distributed as copies of the master database, services, on the other hand, provide the access directly to the master database. Services are also much more flexible in implementing access restrictions, have clear advance in cases of frequent updates (Central server has always the newest data, sending files to all users can create a situation where someone (unintentionally) uses older data.) and can take GIS into wider audience (interacting with spatial services' clients in given DCP is usually quite simple). On the downside they usually require on-line network connections (sometimes considerable amount of network bandwidth is also needed) and are somewhat harder to implement.

I would like to stress once more the importance of open standards, defined by international standard bodies. These standards are based on the consensus of all participants and, if all parties play fair, all three major groups benefit from such standards: governments (regulators can back the 'common goods' for their communities), companies (providers have more opportunities for promoting their products and services) and individuals (users are not locked to systems of only one vendor). Today, every technology provider (company) can participate in most open standardization processes, either directly in consortia (e.g. World Wide Web Consortium (W3C), OGC), or indirectly, through appropriate national standard body, in organizations like ISO. Formally adopted standards also help to prevent the abuse of the power by the creators of the *de facto* standards (such as the enforcement by Unisys of the Lempel Ziv Welch (LZW) data compression patent used in Graphic Interchange Format (GIF) images), as participants often have to identify their intellectual property claims during the standardization process [6, 26].

The need for open standards is perceived not only in GIS community. For example, the electronics design companies, which, very much like GIS system vendors, rely on intellectual talent and resources, are co-operating in developing new, open standards that benefit everybody in the industry [16]. They feel that only open standard with wide (unlimited) range of participants will result in the best, most comprehensive specification. No man is an island and no company can claim all the knowledge for producing a generic solution. The situation exaggerates in the field of communication, diverse set of participants result in a standard that works well with many systems and provides greatly increased interoperability and efficiency.

Those internationally adopted standards provide a great base for national standards (or specifications for a specific application field) after necessary customization [21].

3.1 File Exchange

There are lots of different file formats developed for GIS data over the years. Here is given only a short sampling of some spatial data transfer file formats with the emphasis of

[2] Also called *Distributed Computing Platform (DCP)* [59].

differences between them. Only vector file formats are under investigation in this section (although some of the mentioned formats also support raster data model).

Historically, quite a lot of spatial content has been created with Computer Aided Design (CAD) systems. Although there are similarities between GIS and CAD systems there are also differences that prevent using CAD software in geomatics (and *vice versa*). The main reason is that the GIS data is essentially different from typical CAD drawing. The data used in Geographical Information Systems is composed of high volume of irregular components of small number of primitives, often having fractal structure, while CAD drawings contain lot of regular structures of many (often parametric) types, resulting in a much smaller dataset. The primary data capture methods are also dramatically different (digitizing versus drafting) [40].

Following sections give an overview of some common exchange formats for spatial data. Environmental Systems Research Institute (ESRI) shapefile is a minimalist GIS format that does not support topological relations between objects nor does it store metadata about the dataset. MapInfo data files (*tab* and *mif/mid* files) are also instances of this category.

MicroStation *dgn* is an example of a CAD file format, Another CAD file format that is also used for maps and GIS data is Drawing Interchange File (DXF) from AutoCAD.

S-57 and Digital Geographic Information Exchange Standard (DIGEST) represent the exhaustive spatial data formats that can store a wide variety of geographical data. These formats support topology, metadata, data dictionaries, and other important elements. Both formats are created by international standard organizations and are aligned to the same structures and content when possible. From similar entities SDTS can be mentioned (actually both S-57 and DIGEST are built using the knowledge from the creation and using of SDTS format).

Geography Markup Language (GML) is an exemplar of new, Extensible Markup Language (XML)-based languages that work well with many XML-aware applications for spatial and aspatial processing.

3.1.1 ESRI Shapefile

Shapefile is a spatial data format developed in ESRI, originally designed for ArcView 2 in the early 1990s. The specifications of ESRI shapefiles are freely available from the ESRI Internet site[3] [12, 54].

A dataset in shapefile format consists of at least three files: a main file (*data.shp*) where the actual data is stored, an index file (*data.shx*) and a database (*data.dbf*) for feature attributes. The database is stored as a table in dBASE format. Index file contains offsets of each feature in main data file, thus allowing random access to data. The dataset may accommodate additional files (*data.sbn*, *data.prj* and others), which contain additional information about the data (e.g. the co-ordinate system information is held in *data.prj*).

Shapefiles do not contain any hints about the visual representation of data in files.

The main file of ESRI shapefile is composed of a header and variable length data records, each representing one map feature (*shape*). The header contains information about the dataset—shapefile specification version (currently 1000), bounding box, number of records, and the type of records. One shapefile can currently contain only one type of features. Specified shape types are [12]:

Null Shape A shape with no geometry. Null shapes can be used in the same file with other types of shapes.

Point Simple point in two-dimensional plane, represented by its X and Y co-ordinates. All co-ordinates in shapefiles are double-precision floating point numbers.

[3] http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf

PolyLine A polyline is an ordered set of vertices, representing a polyline (zigzag). Polyline can have multiple parts (a *part* is an ordered set of two or more points). Polyline is located in the XY plane.

Polygon A polygon is a two-dimensional set of one or more rings. A *ring* is a connected sequence of four or more points, forming a closed loop (no self-intersections are allowed). A polygon in shapefile may contain multiple outer rings with each one having possibly 'islands', represented by inner rings. The interior of the polygon is defined by the ordering of points in rings (clockwise for outer and counterclockwise for inner rings).

MultiPoint A set of points which are not connected. Each point is represented by its X and Y co-ordinates.

PointM A 'measured point'—point with X and Y co-ordinates, having also the *measure* M.

PolyLineM A polyline with measure attached to each vertex.

PolygonM A polygon with measure attached to each vertex.

MultiPointM An unordered set of measured points.

PointZ A point in three-dimensional space (X, Y, and Z co-ordinates) with the measure M (total of four parameters).

PolyLineZ A polyline with measures in three-dimensional space.

PolygonZ A polygon with measures in three-dimensional space.

MultiPointZ An unordered set of points in three-dimensional space, each having a measure M attached.

MultiPatch A Multipatch is a set of *patches* (surfaces). A patch inside a multipatch can be either a triangle (several patches form either a *strip* or *fan*) or a regular shapefile polygon. Each vertex has three co-ordinates and a measure M.

It is not possible to store topological information in shapefiles. However, it is possible to compute the adjacency of shapes and build topology for a dataset if certain prerequisites are met (e.g. there are no gaps or overlaps between polygons which should be adjacent). The absence of topological information and relative simplicity, compared to other GIS file formats (including ESRI's own *coverage* format), gives usually faster access times for reading data from files [54].

Every feature (shape) in an ESRI shapefile can have associated attribute data, which is stored in the dBASE file (one row for every feature, identified by unique feature identifier). Attributes can have various data types (strings, integers, floating point numbers, etc.), according to dBASE file specification [5].

ESRI shapefiles are nowadays read and written by many applications, including all major GIS programs, and the format has gained popularity, becoming one of the *de facto* standards for spatial data interchange. Its simple structure is a bonus in cases when topology is not needed, but the lack of accompanying metadata would lead to not to recommend its usage between systems when there is slightest possibility of misunderstanding.

3.1.2 Intergraph/Microstation Design File

The Intergraph Standard File Format (ISFF) (*V7*) is a CAD file format which is also quite popular for storing geographical data, especially in the case the data originates from a CAD-based GIS. The ISFF (also known as the *dgn* format, as the ISFF filenames have usually a suffix '.dgn' appended) is the file format common to MicroStation[4] and Intergraph's Interactive Graphics Design System [38].

The *dgn* file was designed primarily for storing CAD drawings and its history is clearly visible from the file structure. ISFF are binary files, either *design files* (common suffix (filename extension) '.dgn') or *cell libraries* (suffix '.cel'). Design files contain both graphical and non-graphical (invisible) elements. Cell libraries contain a series of cells—complex elements used for (multiple) placing into design files.

A graphical element is a representation of some constellations (lines, ellipses, etc.) by Cartesian co-ordinates (co-ordinates are stored as 32-bit integer numbers) and can be a simple (e.g. line or cone) or complex element (a conglomerate of several simple elements). Both 2D and 3D elements are allowed. Each element in ISFF is specified to be on one of the 63 *layers*. Some of the graphical elements:

Line A line segment, represented by two points.

Line String Three or more connected, ordered points, forming a line string (zigzag).

Point String is a set of ordered points which are not connected (previous points define orientation at each vertex).

Shape is polygon, formed by a closed line string (the first and last points of the series must be coincident)

Curve Parametric spline curve, defined by a set of points. The curve passes through all these points.

B-spline Curves and surfaces defined by B-splines (instead of linear interpolation between vertices). Design files can contain rational, non-rational, uniform and non-uniform B-splines.

Text Text with fine-tunable parameters.

Complex Chain is a chain (closed line-string), formed from several simple elements.

Complex Shape is a shape, formed from several simple elements.

Ellipse An ellipse is defined by its major and minor axes, center point co-ordinates and rotation.

Arc An arc of an ellipse.

3D Surface is a complex 3D element that is projected or rotated from a planar boundary element (line, line string, curve, arc, or ellipse). Surface is capped at both ends, enclosing a volume.

3D Solid is similar to surface, but it is not capped, thus enclosing no volume.

Cone A circular cone, truncated cone or cylinder, possibly skewed.

[4]MicroStation is a technical CAD system, a product of Bentley Systems, Inc.

Raster Data This element defines a piece of raster data to be inserted into the design.

The file format from *MicroStation V8*, released in 2000, is a substantially different file format, although the files carry the same suffix (.dgn). The new file structure leverages many problems with the old one, such as the the allowance of infinite number of levels [7]. The specifications of this format are not published.

Every graphical element in ISFF carries information about its presentation. This information includes the *color index* (actual colors that correspond to indices (total of 255) are also defined in the design file), *weight* (thickness of lines), *style* (solid, dashed, etc.) and several different *properties*. Symbols are usually constructed from the same elements as other drawings and are stored as cells (complex elements) either in design file or in a separate cell library [57].

ISFF does not contain, similarly to the shapefiles, any means of storing explicit topological relations in the data file. The most one can do with *dgn* files is to ensure the correctness of data (that means, for example, that coincident lines are really coincident in all points) just before the process of data exchange, because spatial datasets are very fragile and editing is error-prone without the help of topology.

The design file format does not let the user to store metadata in the file, thus severely crippling data exchange using ISFF (There is a place for storing co-ordinate units and user can make linkages to outside databases, but using these linkages result in an system-specific behaviour or total mess in the worst case.). This fact together with the lack of clean separation between the data itself and its representation leads to the conclusion that ISFF (*dgn*) is not appropriate file format for digital spatial data exchange between different systems.

3.1.3 S-57

The *S-57* (IHO Special Publication No. 57) is a standard to be used for exchange of digital hydrographic data (also known as Electronic Navigational Chart (ENC)) between national mariners, hydrographic offices, and other users [25].

The full title of the S-57 is *IHO Transfer Standard for Digital Hydrographic Data*, the effective version of this specification from November 2000 is Edition 3.1. The text of the standard is available from IHO for 150 € [24].

S-57, originally developed as a profile of the SDTS, called *DX-90*, is an official IHO standard for nautical charts, all hydrographic offices worldwide are required to produce digital charts in this format.

S-57 is intended purely for data transfer, real applications are expected to convert the ENC data into vendor-specific (System ENC (SENC)) format that is better suited for displaying and other tasks in GIS (differences between ENC and SENC formats are usually syntactical, the meaning of data is the same). This is in contrast with previously described proprietary file formats which were used mostly as a ‘data store’ for certain applications (The ability of using these files for data exchange was an included benefit that was probably not a high priority item in the head of the creators of those formats.).

SENC data is most often used in specialized GIS called Electronic Chart Display and Information System (ECDIS). ECDIS is a complex system, installed on the bridges of ships, that helps mates in navigating the vessel. The system displays electronic charts with the data from several sensors (position, speed, surroundings, etc.) and miscellaneous supplementary data (e.g. official notes to mariners and captain’s own notes). S-57 has no concept of presentation, leaving the visualization of chart content to the applications. In ECDISs the presentation model defined by S-52—*IHO Specification for Chart Content and Display* *of ECDIS* is used (other applications can use different presentation models). If the ECDIS follows the *International Maritime Organization (IMO) Performance Standards for ECDIS* then it can be used in place of traditional (paper) charts (The use of S-57 and S-52 standards is mandatory by the IMO specification) [1, 23, 48].

As a transport form for digital data, the creators S-57 have paid a lot of attention to the process of updating nautical charts. Updates are given in the form of differences against the base chart, i.e. there are only inserted, deleted or modified records in the ENC update file. Those arrangements make easy updating possible even over slow communication channels (like radio links on board of ships). Checksums are used for all datasets to ensure the integrity of data during the transfer.

Metadata is quite well handled, each chart has a fixed structure in its heading with fields for data producer information, parameters of the dataset (accuracy, source, topological level, and others) and geodetic data (used projection, datum, co-ordinate units, bounding box co-ordinates, etc.).

The S-57 data model is composed of feature objects and spatial objects. *Feature object* (feature) is abstraction of a real-world phenomenon, it contains descriptive attributes but has no geometry. Feature objects are located by spatial objects. Each feature object has an associated *geometric primitive*, indicating whether the object's type is *point*, *line*, or *area* (objects which do not reference any geometry are also possible). Areas may have insets ('islands'). Similar feature objects belong to the same *class* (e.g. roads or wrecks). Classes are defined in the *data dictionary*.

Spatial object contain the geometry, although it may also contain descriptive attributes. Vector geometry is currently the only available form of spatial information that can be coded in S-57.

All co-ordinates are given in two- or three-dimensional space. Three-dimensional co-ordinates are only used for sounding points (feature class `SOUNDG`)[5], for all other objects the third dimension is expressed, when necessary, as an attribute of an object (e.g. the height of a lighthouse). Locations of points can be expressed in planar or in geodetic co-ordinates. Vertices are connected either by straight lines, elliptical arcs, Bezier, or B-splines.

The S-57 format can store topological information and therefore it is often not possible to describe the geometry as one spatial object but the geometry of one feature object have to be broken up to 'bits and pieces' with additional relations between them.

A spatial object is either a *isolated node* or *connected node* (representing a point), an *edge* (representing a line), or *face* (representing an area). There are four topological models defined in the standard which use different relations on different types of spatial objects for encoding the geometry:

Spaghetti is a data structure where all lines and points are unrelated to each other (no topological relations are stored). *Edges* and *isolated nodes* are used, areas are represented by edges that track the perimeter of the polygon.

Chain-node is a geometrical data structure, composed of *edges*, *isolated nodes* and *connected nodes*. Edges and connected nodes are topologically linked. Lines are composed of edges which reference a connected node at each end, areas are represented by edges that track the perimeter of the polygon.

Planar graph is a two-dimensional data structure in which the geometry is described in terms of *nodes* and *edges* which are topologically linked. It is a special case of a *chain-*

[5] Soundings have also exceptional feature object type—they may have an arbitrary number of unrelated co-ordinate points (multipoint geometric primitive), while formally being point objects.

node structure where edges are not allowed to cross (a connected node is inserted to all junctions).

Full topology is a two-dimensional data structure in which the geometry is described in terms of *nodes*, *edges* and *faces* which are all topologically linked. A *planar graph* with faces.

The connection between feature and spatial objects depends on the topological model used: for *spaghetti* model there is one-to-many relationship, any objects with coincident geometry (like land area and coastline) results in duplication of spatial data. Other topologies do not allow the duplication of co-ordinates and many-to-many relations must be used. It is also possible to define feature collections (relationships between features).

One important part of the S-57 standard is the *data dictionary.* Data dictionary is part of the metadata for ENC, it describes feature classes, attributes of features and spatial objects and relations between objects and attributes (which attributes are allowed for each object class). A class has a name, type (geographic, cartographic, meta, etc.), numerical code, short and long descriptions. Attributes have also name, code, descriptions, but also type (integral, real, string, enumerated) and domain (a set of permitted values, specified either by enumeration or by boundaries). Data dictionary is transferred from data providers to users like other datasets (either in full length or as updates), and is then installed into user's GIS (ECDIS or other). Individual ENC datasets do not include the data dictionary.

S-57 uses the ISO/IEC 8211 standard (*Specification for a data descriptive file for information interchange*) for encapsulating digital hydrographic data. That standard provides a mechanism for transferring digital data (in the form of files) from one computer system to another, independent the make and characteristics of such systems, or the medium used for transmission.

ISO/IEC 8211 uses so called *data descriptive records* for describing the logical structure of the actual data contained in the file. With such schema it is possible to decode the file with no *a priori* knowledge about the structure of the data file (field names, lengths and data types).

S-57 has an appendix called *ENC product specification* which refines and clarifies the main part of the standard. This appendix also imposes major restrictions to all conforming S-57 datasets: all charts must use chain-node topology with linear interpolation between the points, all points must be in geodetic co-ordinates (using the WGS 84 datum), textual data has to be in English, and so on [25].

3.1.4 DIGEST

The *Digital Geographic Information Exchange Standard (DIGEST)* is developed by DGIWG. The most current version of this standard is edition 2.1 from September 2000. DIGEST is an open standard, its text is freely available to everybody at the DIGEST Internet site[6]. The purpose of this standard is enabling the seamless transfer of digital geographic information between Geographical Information Systems. DIGEST supports the exchange of both raster and vector data. Standard specifies data structures, means of data encapsulation and the data dictionary (*Feature Attribute Coding Catalogue (FACC)*) used for describing the dataset [9].

The DIGEST format resembles, at the conceptual level, the United States' Spatial Data Transfer Standard (SDTS). It is built in close co-operation and alignment with other international standards, such as IHO S-57, a suite of geomatics standards developed by ISO/TC 211, etc.

[6] http://www.digest.org/

The dataset can be encapsulated in a number of ways: ISO 8211, generally used for bulk transfer and for archival purposes, ISO 8824 (Abstract Syntax Notation number One (ASN.1)) used in telecommunication, Vector Relational Format (VRF) for database user applications, and Image Interchange File (IIF) for image and raster data. DIGEST defines also media and labeling standards for magnetic tape, optical discs, and other exchange media. In addition, file naming conventions and character representation are also specified.

Datasets that are formed in accordance with DIGEST guidelines can contain topological information. DIGEST supports vector data with four levels of topology and also raster data. The topological model and levels are identical to those used in S-57 (Minimum Redundant Topology). Vector features are classified as *point*, *line*, *area*, *text*, or *complex* (containing other features) features. Spatial extent of features is expressed in *node*, *edge*, and *face* elements (these elements can carry positional attributes (e.g. about the measurement accuracy of co-ordinates of a given point)). Points are specified in three-dimensional space. Raster data files contain pixels whose value (colour) is defined as a red-green-blue triplet or by using look-up tables for getting the correct colour value. Co-ordinates for registering the raster image are also stored with the image.

Information about attributes and feature classes is kept in a data dictionary (Feature Attribute Coding Catalogue (FACC)). Each feature object is identified by a code which also identifies its category (culture, vegetation, etc.).

Extensive metadata is included with vector and raster datasets, containing data source information, projection and reference points for scanned raster maps, various data quality indicators, such as completeness and consistency levels, accuracy, and much more.

The DIGEST specification has been customized to form a national standard for cadastral data [21].

3.1.5 GML

Geography Markup Language (GML) is an XML encoding for the transport and storage of geographic information, including both the spatial and aspatial properties of geographic features. The GML specification (current version is 2.1.1 from 25 April 2002) is prepared by the Open GIS Consortium and can be found at the OGC site[7]. Current status of the GML standard is an *OGC Implementation Specification*.

The purpose of the GML standard is to provide an open, vendor-neutral framework for the definition, storage and transport of spatial application schemas and objects for specialized domains, to encourage the sharing of application schemas and the information they describe.

Geography Markup Language is designed to separate the content cleanly from its presentation and the standard only specifies the encoding of spatial data without any notices to the visual representation. The wide range of XML tools can be used to transform the data in GML to any visual form (raster or vector graphics, voice, etc.) [20].

GML has great impacts on the systems that deliver spatial data over the World Wide Web (WWW). Typically, maps in the WWW environment used to be synonyms to raster images (GIF, Portable Network Graphics (PNG), etc.). GML brings a new dimension into the spatial data exchange via Internet with the separation of the content of the map cleanly from its presentation. The visual rendering is done on user's side, with user-specified tools and parameters. Also, maps, encoded in GML are 'true' (vector) maps, supporting queries and editing, enlarging the possible audience. There are also an increasing number of tools that can be used for processing GML data as the XML technologies make steady progress in the WWW environment [17].

[7] http://www.opengis.net/gml/02-009/GML2-11.html

A GML document is described in a metalanguage called *XML Schema* [62]. The current GML standard specifies three base schemas: Geometry schema for encoding geometry, Feature schema for describing basic properties of features, and XLink schema for making links between objects, these three are normally referenced in user-defined schemas. Users of GML are expected to create new application schemas for their application domains, based on the three schemas provided by OGC.

The standard allows (and encourages) the creation of *profiles*—subsets of the GML standard that fit some specific application domain. An application schema is then created from the basis of a given profile.

The data model of GML is based on the *OGC Abstract Specification*[8] [52] which provides the reference model for Open GIS Consortium Implementation Specifications. A dataset which represents the real world is defined by a set of *features*, each having properties (attributes). Some properties of geographic features describe its geometry. Features can be aggregated into feature collections.

GML operates with with the simplification of the OGC Abstract Specification called *simple features* where the geometric properties of the features are restricted to 'simple' geometries for which co-ordinates are defined in two dimensions[9] and the delineation of a curve is subject to linear interpolation.

The Feature schema defines a small set of geometric properties for association of geometries (defined in the Geometry schema) with features. Examples of these properties include *location*, *centerLineOf*, and others.

The Geometry schema describes the primitive geometrical constructs that form the basis for spatial databases. The current version supports only vector data (this restriction results from the *simple features* specification). All co-ordinates can be specified in two- or three-dimensional space. Each element may have an optional pointer to a co-ordinate reference system. All elements of geometry collections (MultiPoint, MultiLineString, MultiPolygon, Multigeometry) must share the same spatial reference system, which is included only for the collection element. The Geometry schema defines following geometrical elements ([20]):

Point element represents a single co-ordinate pair (triple).

Box is a rectangle, represented by two co-ordinate tuples (the minimum and maximum values along co-ordinate axes). Box elements are used for denoting extents.

LineString is a piece-wise linear path defined by a list of at least two co-ordinates that are assumed to be connected by straight lines.

LinearRing is a closed LineString where the last co-ordinate tuple must be coincident with the first one. LinearRings are used for building Polygons.

Polygon is an element representing a connected surface, made up from LinearRings. A Polygon can contain 'islands' (holes) inside the main area.

MultiPoint is a collection of Points.

MultiLineString is a set of LineStrings.

MultiPolygon is a collection of Polygons.

[8]http://www.opengis.org/techno/abstract.htm

[9]GML 2.1 includes some rudimentary support for three dimensions (co-ordinates of a point can be specified in 3D space) but no geometrical constructs are allowed to use the third co-ordinate.

Property	SHP	ISFF	DIGEST	S-57	GML
Metadata	no	no	yes	yes	yes
Co-ord. system	yes*	no	yes	yes	yes
File type	binary	binary	binary	binary	text
No. of files	many	varies	one	one	varies
Topology	no	no	yes	yes	no
Presentation	no	yes	no	no	no
Company controlled	yes	yes	no	no	no
Specifications	free	rev. eng	free	non-free	free

For explanations see the text of section 3.1.6.

Table 3.1: Comparison of selected GIS file formats.

MultiGeometry is a container that can hold arbitrary geometries (primitive and/or collection elements).

Version 2.0 of GML does not support topology, although future versions are expected to have this capability. The base schemas provide very few hooks for attaching metadata to the spatial dataset (e.g. names of the features, spatial reference systems for geometries). At first glance it may seem not appropriate for today's systems where metadata is given an ever increasing rôle. When one dives deeper into the GML standard it becomes clear that by the creation of application schemas everyone can adopt any particular dataset to have exactly the amount of additional (meta) information one believes sufficient. GML with accompanying schema definitions is perhaps the most flexible format available today for encoding spatial data.

3.1.6 Comparison of File Formats

Different spatial data exchange formats possess quite diverse characteristics as they are created with different tasks in mind. If one is about to choose the file exchange format for spatial data he may look at some crucial properties of file formats that were covered in previous sections, as summarized in Table 3.1.

The rest of this section contains explanations to the table.

Metadata Whether the format supports metadata (remarks about the main data) in the dataset.

ESRI shapefiles and ISFF files do not provide any means for storing metadata in the data files, users must put the accompanying metadata (e.g. data source and accuracy) to free-format files and take care of distributing them together with data files.

DIGEST and S-57 provide predefined fields for storing metadata. This indeed encourages the using of descriptive (meta-) data but unfortunately also restricts the user to predefined set of values. However, in most cases the standardized form for metadata is sufficient.

The most flexible format is GML which puts the full weight to the shoulders of implementors of a profile (sub-format). The GML specification describes only the coordinate system metadata (see next item), the profiles are free to define and use whatever descriptive data is needed for given applications.

Co-ord. system The possibility to attach co-ordinate system info (positional metadata) to actual files. This kind of metadata is essential for interpreting the co-ordinates correctly. Unfortunately the DGN files do not contain the place for such information.

ESRI shapefiles have a place (*prj* file) for storing co-ordinate system data but a large number of datasets do not include this information (this is a later extension to the file format specification).

File type The type of files, either binary or text. GML, like all XML formats, is a text-based file format while the other formats covered here apply binary encoding on the files. Text files have their advantages and disadvantages, the main benefit being the readability by humans as well as by computers, providing the ability to check or modify the file 'on-the-fly' using only a text editor.

On the other hand, they take a lot more space on storage media and require more processor power for parsing than their binary counterparts. For example, a simple GML file, containing only one polygon feature with 177 vertices on the outer ring and 7 on the inner ring, and having 13 attributes takes as much as 20 KB of storage. The binary representation of the same data would need a fraction of this space.

No. of files Number of files in typical dataset (here we do not count in the data dictionaries, presentation libraries, style-sheets and other auxiliary files). S-57 and DIGEST provide the whole dataset in one file while the shapefiles require the division into multiple files (The restriction of only one type of shapes in one file, one may also want to separate different layers (with different attributes) into separate files.).

The DGN files support all graphical elements in one file so the dataset can be stored in one file. Sometimes the dataset is broken into number of files according the layers of the drawing or element types.

The GML does not make any assumptions about the number of files and leaves the problem to the users. While it is possible (and often most practical) to put all data into one file it is also legal to break the dataset into smaller pieces (GML files tend to grow quickly in size). Referencing between those pieces is supported.

Topology indicates the the ability of storing topological relationships between features in given format. S-57 and DIGEST are the only files from the that support topology.

Storing the geometry in topology-aware data structures should be essential for all non-trivial spatial databases and the support of topology should be considered mandatory for formats, used for data exchange for 'solid' and 'serious' tasks.

It is important to notice the different topological models that can be used in these files. Other data exchange formats not covered here are known to use yet different models which may complicate the data reuse (conversion between topological models is needed).

Presentation info Whether the information about the visual appearance of data (line width, symbol shapes, etc.) is described in the same file. Only the ISFF stores presentation integrated into the data file (like other CAD systems do). Such tight coupling makes the change of presentation very painful.

The opposite practice of making separate files for instructions that translate spatial data into meaningful visual appearance is preferred in most situations, as such presentation file can be used for many datasets and they can be changed quickly in case different symbolization is needed.

Company controlled indicates whether the file format is controlled by one corporation or not. Clearly the data exchange formats that are not driven by one company but an international standard body or consortium are preferred, as the latter include the wisdom from larger number of people and are generally designed to interoperate better with different software products.

The individual corporations (however big they are!) tend to be more volatile than consortiums and either stop supporting the products that handle the specific data formats or even cease to exist, leaving its customers dead in the water.

Specifications Whether the specifications of given file format are available gratis (*free*) or for a nominal fee (*non-free*). The specifications of the proprietary file formats are often not available (to wider audience) at all. This used to be the case with DGN format but its specification is disclosed lately ([38]). However, this specification does not provide full description of all the fields in the file, the meaning of which is being reverse engineered by volunteers [56].

3.2 Services

The ISO standard on the terminology of geomatics gives the following definition for the term *service*:

> 'A capability which a service provider entity makes available to service user entity at the interface between those entities [19].'

It can be seen from this description that the service provider and the service user work together for the process of data exchange and users get on-line access to the data. This is in contrast to file exchange where the data source (provider), more often than not, is independent from the data consumers and work is done off-line.

Services are implemented on top of *communication protocols* (frameworks) which implement the low-level primitives for connecting two applications. In distributed environments the communicating processes can reside anywhere in the network. Popular platforms used nowadays include HTTP (the foundation behind the now-ubiquitous World Wide Web), CORBA (a broker-based distributed environment), .NET (a new generation XML-based DCP) and others.

Some frameworks (e.g. CORBA) provide broker services for locating service providers or consumers, while others (e.g. HTTP) do not and the task of locating clients, servers or peers in the network is left completely to users of the protocol.

A client can connect to many services at the same time using different communication frameworks, as shown in Figure 3.1 (technically, the client application on the figure has to implement two separate client interfaces for the two DCPs).

The purpose of this section is to give an overview of some services that provide spatial context to their users. Due to the great variance in available services and the limited space only web-based services (using the HTTP for data transfer) are discussed here.

Most attention is given to the standards developed by Open GIS Consortium. These specifications represent the first attempt to standardize the field of spatial web services. As the OpenGIS standards are open and well-thought, they will surely get wide adoption in the nearer future. When the OpenGIS infrastructure is universally accepted it becomes easy to implement various types of services providing geospatial content that plug seamlessly into the network of other similar services and become instantly accessible to the users.

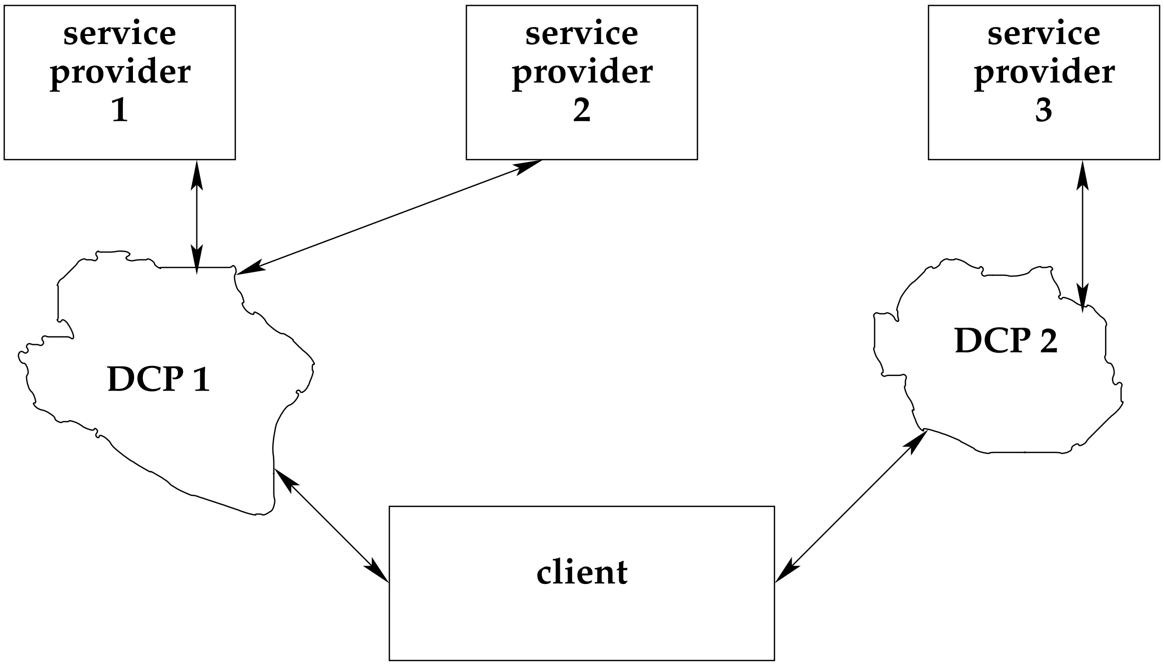


Figure 3.1: The overall concept of services.

Standardized web services will lead us to unification, increased interoperability, and wider utilization of spatial information and services. Using web services for accessing geo-spatial data can, in longer run, provide a whole environment for miscellaneous web-based Geographical Information Systems—WebGIS. A *WebGIS* is a complex system with access to the Internet for capturing, storing, integrating, manipulating, analyzing and displaying data related to locations without the need of having any proprietary GIS software [43].

3.2.1 Proprietary Map Servers

The first web site that included geographical maps was established by Xerox Palo Alto Research Center in 1993 [46]. Today most big GIS software vendors have developed some forms of web services (usually in a form of a *map server* which creates raster images based on the data layers and location, specified by the client.). There are wide variety of such servers (see, e.g. an article by F. Limp ([31]) for comparison of various map servers) that differ in functionality, methods of user interaction (client interface) and price. Unfortunately there is no interaction between those web services, a user cannot combine the results (e.g. raster images of maps) of different services.

An example of a proprietary map server is Maptech *MapServer* (Not to be confused with the MapServer from the University of Minnesota, covered in section 4.1.), which can be found at `http://mapserver.maptech.com/homepage/index.cfm`. The server provides map images for locations that can be specified by co-ordinates or, using the geocoding capabilities of the server, by postal codes or city names. The geocoder works only for the addresses inside the USA, co-ordinates can be specified for all other places as well. Four kinds of maps are provided: topographical, nautical, aeronautical and aerophotos (not all of these are available for all regions). Maps are rendered as Joint Photographic Expert Group (JPEG) images.

The client side is a simple application, composed of Javascript and Hypertext Markup Language (HTML) that runs in a web browser. Zooming (a few predefined values) and panning are supported, geographical co-ordinates are provided for map center. An interesting feature of Maptech MapServer is the ability to insert and label private points of interest on the map. The label is composed of a text (possibly containing a hyperlink) and a symbol from a large array of predefined symbols.

Standards are emerging to remedy the previous 'disastrous' state of the World Wide Spatial Web [22]. Existing services often try to give as many attraction as possible, stuffing many functions into one interface (Maptech MapServer) without clean separation between them (the output of a geocode is displayed as a map, without any possibility to get explicit co-ordinate information, for example). Formal standards, developed by OGC (covered in the rest of this chapter), specify clean interfaces for every functional unit (e.g. geocoder, co-ordinate transformation server, map server, etc.) that are able to work together but are perfectly usable individually.

3.2.2 OpenGIS Web Services

The OpenGIS Consortium, being an international industry-wide organization, has created a set of high quality specifications for different types of spatial services. This section gives an overview of the OpenGIS Web Services and outlines the common components of these services.

The OGC Web Services are divided into two groups: *OpenGIS Web Mapping Services* and *OpenGIS Geospatial Fusion Services*. OpenGIS Web Mapping Services include standards that make possible the dynamical querying, accessing and processing of spatial information in the WWW environment. The parts of the Web Mapping Services family include WMS, WFS, Web Registry Service (WRS), and Web Coverage Server (WCS).

OpenGIS Geospatial Fusion Services deal with the non-map information that refer to places[10]. Such information can be in various formats: text, video, photographs, and so on. These services support different activities, such as searching, discovery and sharing of spatial information, standards of this group handle different aspects of spatial information depending on the content and purpose of the data (addresses, place descriptions, etc.). The Geospatial Fusion Services include *Gazetteer Service*, *Geocoder Service* and *Location Organizer Folder* and others [41].

This work concentrates to the first group of standards (Web Mapping Services), a longer overview is of these and other OpenGIS standards is included in the paper by Kuus, Krusberg and Rebane ([29]). The next sections describe some selected standards from the OGC Web Mapping Services.

Common Elements in OpenGIS Web Services

This division discusses some bits and pieces that are common to all web services as specified by Open GIS Consortium.

All OGC web services use the HTTP as an underlying transport protocol, there are also expansion hooks for adding support for other DCPs in the future. The communication between the server and client is carried out by *requests*, which are sent by the client to the server using platform-specific techniques (e.g. HTTP GET method). The server then returns the appropriate answer (map image, grid coverage, error message, etc.) after it has finished processing the request. Each request contains a number of *parameters* whose names and values are defined in appropriate standards. Software vendors may include additional *vendor-specific parameters* which introduce additional possibilities to the systems that are aware of them (usually the systems of the same software vendor).

[10] I.e. the spatial information that is not expressed explicitly in co-ordinates but in some other form instead, like a postal address.

Every compliant web service has to support the *GetCapabilities* request which returns description about the service, its information content, a list of supported protocols, requests and parameters. This information is provided to the clients of the service in XML form, providing easy parsing by computers as well as by humans. One physical server may be able to simultaneously act in place of several logical services (e.g. WMS and WFS), in this case the server capabilities document will contain information about all these services. The *GetCapabilities* request is supposed to be the first contact between a server and a client in the session. During this request the version negotiation is carried out when necessary.

All OGC web services support step-by step negotiation of version of a service protocol that will be used between the service provider and user. The process of negotiation is carried out as a part of a *GetCapabilities* request: if the version in the client's request is not supported by the server it responds with another version number which is either accepted or rejected by the client. In the iterative process of negotiation the server and the client will come to an agreement over the exact version of the protocol that will be used for further communication (if they cannot find a common subset of the protocol the following actions would be meaningless).

Whenever a server encounters a situation where it cannot recover from (i.e. invalid parameter value supplied by the user), it throws an *exception* back to the client where the request came for. Common XML-based exception formats are defined for all OGC services, there may be other formats available for reporting errors (e.g. WMS standard defines exceptions in the form of raster images). Client can specify a preferred exception format in its requests.

3.2.3 OpenGIS Web Map Service

The *Web Map Service (WMS)* is an Open GIS Consortium standard for a service that produces georeferenced maps in the WWW environment. The full title of this normative document is *Web Map Service Implementation Specification* and it is a OpenGIS Implementation Specification, currently at version 1.1.1 [59]. The specifications are available for free download on the OGC web site[11].

The WMS standard is the most mature of all OGC web services, it also formed the basis for the ISO standard ISO/WD 19128 *Geographic information—Web Map Server interface* (currently in the status of a working draft), being developed by the ISO/TC 211 (*Geographic information/Geomatics*). The ISO standard is meant to be compatible with the OpenGIS Consortium specification.

WMS builds on top of the HTTP protocol and uses a HTTP server as a mediator between WMS server and its clients (Figure 3.2 on the following page).

The WMS protocol specifies three requests: *GetCapabilities*, *GetMap*, and *GetFeatureInfo*. An example of a session between the server and a client is shown in Figure 3.3 on the next page (real clients would probably make several *GetMap* and *GetFeatureInfo* requests during a session).

An example of the WMS *GetMap* request for drawing three layers: `http://195.50.203.246/F-Open-server/server.php?VERSION=1.1.0&REQUEST=GetMap&LAYERS=Base-NC610,Navigation-NC610,Lights-NC610&STYLES=&SRS=EPSG:3300&BBOX=551122.3,6592254.9,556126.4,6599329.6&WIDTH=406&HEIGHT=574&FORMAT=image/png`.

In the context of WMS the map is defined as a visual representation of geodata, a map is not the data itself. This specification defines three WMS operations (requests): *GetCapabilities* returns service-level metadata, which is a description of the service's information content and acceptable request parameters, *GetMap* returns a map image whose geospa-

[11] http://www.opengis.org/techno/specs/01-068r3.pdf

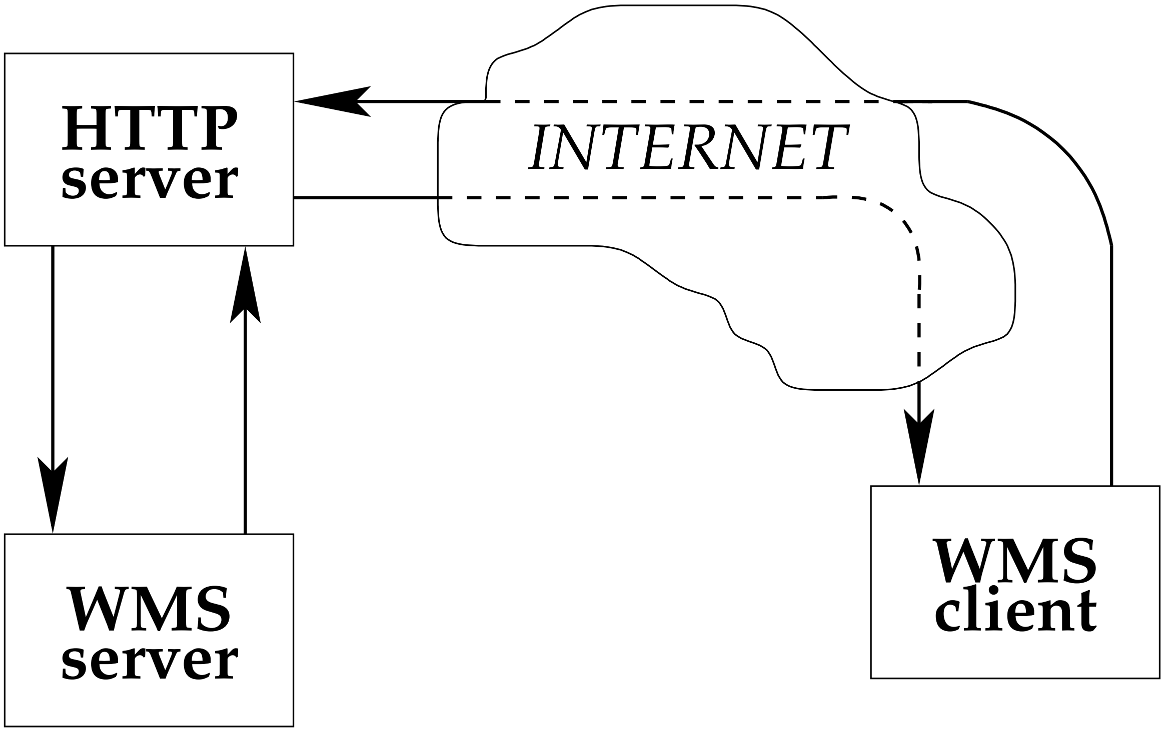


Figure 3.2: Data flow in WMS.

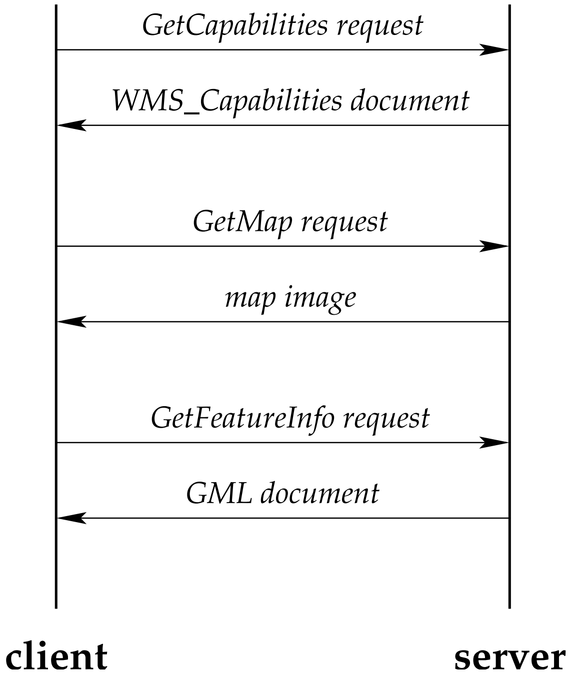


Figure 3.3: A sample session between a WMS server and a client.

tial and dimensional parameters are well-defined, *GetFeatureInfo* returns information about particular features shown on a map. The implementation of the *GetFeatureInfo* request is optional to WMS software and service providers [59].

The data in WMS servers is divided into *layers*. A layer is an atomic entity for the user request (*GetMap*). Some layers may be marked as *queryable*, in this case the clients are entitled to make *GetFeatureInfo* request for these layers. For each layer there are defined the name of the layer (which is used in client's requests), spatial extent (bounding box), a list of Spatial Reference Systems[12], and other parameters, such as additional dimensions, units, pointers to additional information about the layer's data, etc.

For each layer there have to be one or more *styles* that define the presentation of layer's content. Normally the WMS servers use static styles that can be referenced by their respective names. It is also possible to use WMS servers together with the *Styled Layer Description (SLD)* specification that provide the user-defined symbolization for map layers. With SLD the client gets a map image that is visualized using the user-supplied data, resulting in the desired visual appearance of the map [10]. The SLD WMS adds some additional requests to the protocol that are not covered here.

The map image produced by a WMS server is created with well-defined geospatial and dimensional parameters and can be used wherever there is a need for spatial data. Two or more maps can be blended into a composite image when they have the same bounding box, Spatial Reference System (SRS), and dimensions.

The most common form for the map image, generated by the WMS server, is some ubiquitous raster image format—Tag Image File Format (TIFF), PNG, JPEG, etc., but the image can be also in vector graphics format (e.g. Scalable Vector Graphics (SVG)) or in non-graphical descriptive form (GML), although the latter is being largely superseded by the Web Feature Servers. Other data that is occasionally sent from server to its clients is in XML form, making the automatic parsing and understanding it easy for the client programs.

A WMS server can act as a *cascading server*, that is act as an WMS client to other servers while still performing as an WMS server for its clients. Cascading server has to redirect some (or all) of the requests to other servers and send the output returned by those servers back to its client. Such intermediating servers can also perform various 'enhancements' on the data, such as image format or co-ordinate system conversions, or aggregation of data from several other servers.

The WMS server throws exceptions, as all OpenGIS web services, when it cannot further proceed the current query. In addition to the standard OGC XML-based exception format WMS also defines exception indication messages in the form of raster images. Either a blank image or an image with the text describing the situation is sent instead of map image in case of a raised exception.

The requests defined in the WMS specifications [59]:

GetCapabilities The *GetCapabilities* request gives service metadata about the informational content (technical contacts, access restrictions) and acceptable request parameters of the particular server (Uniform Resource Locator (URL)s and file formats). The information about the data (keywords, spatial and temporal extents, SRS, data provider contact, etc.) and its presentation (styles) is provided for each layer. The description is provided in a machine- and human-readable XML form.

GetMap The *GetMap* operation is designed to produce a map, either as a pictorial image or as a set of graphical elements. The resulting map is produced from the user-specified

[12]The WMS specification uses the co-ordinate systems from the European Petroleum Survey Group (EPSG) database of geodesy parameters (`http://www.epsg.org/`) or alternatively by specifying the parameters of a projection (only a few co-ordiante systems are supported using the latter method).

list of layers and styles, using the bounding box, SRS, width, and height supplied in the request parameters.

GetFeatureInfo The *GetFeatureInfo* request is designed to provide clients of a WMS with more information about the features in the maps that were returned by previous *GetMap* requests.

The parameters of this request include the parameters of a *GetMap* request (from this information the WMS server determines the location and scale of the area of interest). Additional parameters provide the layer names to be queried, pixel co-ordinates of the point that caught user's interest and the desired format of returned information.

The server returns the map features that are found in the specified point. A server can report map features in several formats, as described in its response to the *GetCapabilities* request. The GML is often used for this purpose for its fitness for the task (easy processing, tailorable for different application domains, relatively small data sets). The information returned by the *GetFeatureInfo* request resembles that of a WFS (see the next section).

The *GetFeatureInfo* request is acceptable only for layers that are marked *queryable* in the service metadata document.

Some implementations of WMS servers and clients are covered in Chapters 4 and 5. Chapter 6 on page 56 gives some examples of utilizing the WMS protocol for solving real-world problems.

3.2.4 OpenGIS Web Feature Server

While the *GetFeatureInfo* request (indeed all other requests of the WMS specification too) of WMS only supports one-way data exchange between server and client—map features are sent from server to clients for inspection (and possibly for saving to client-side storage) but there are no methods for the client to alter the data in server's database. The WFS specification contains different requests for clients for retrieving the feature data from the server and also modifying the data in server's datastore. WFS is meant to supplement WMS, the two can be coupled very tightly.

The *Web Feature Server (WFS)* is an OGC standard (*Web Feature Server Implementation Specification* ([58])) specifying the handling (querying, updating, insertion, deletion) of map features in spatial databases using the HTTP protocol.

A WFS request contains a description of query or data transformation operations that are to be applied to a web-accessible datastore. The WFS specification is independent of the type of database (Relational Database Management System (RDBMS), XML-files, etc.). All requests (and server responses) are encapsulated into the HTTP protocol, using either *GET* or *POST* methods. The standard form of data exchanged between the WFS server and client is XML (GML for spatial data). Exceptions are raised when the server cannot parse user's input or other abnormal events occur.

The standard defines two classes of Web Feature Server servers: the *Basic WFS*, which would provide *read-only* access to the datastore, and the *Transaction WFS* which provides also full support for transaction-based modifications in the database.

The *Basic WFS* must support *GetCapabilities*, *DescribeFeatureType* and *GetFeature* interfaces, *Transaction WFS* must support these, plus additionally *LockFeature* and *Transaction* requests. A sample session between a server and a client is sketched in Figure 3.4.

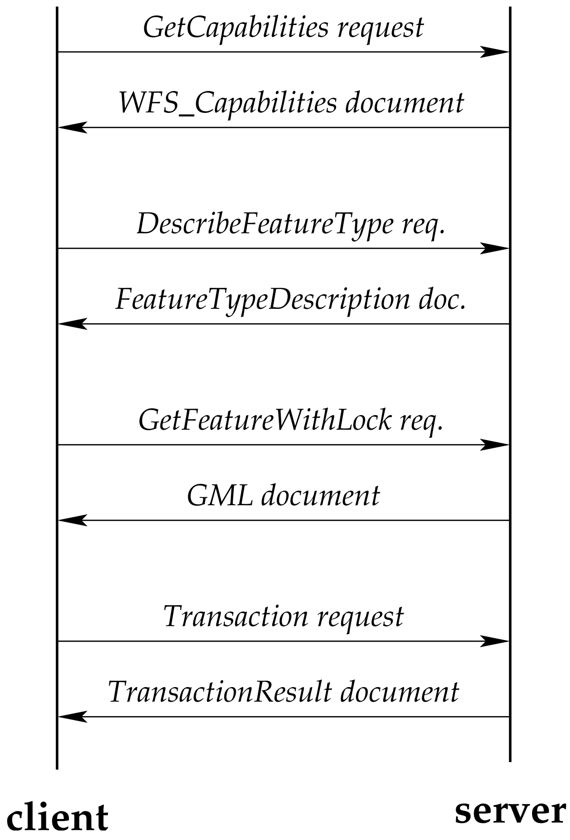


Figure 3.4: A sample session between a WFS server and a client.

Features can be queried either one-by-one or using a filtering system defined in *Filter Encoding Implementation Specification*. The OpenGIS standard for filter encoding provides a flexible system for specifying spatial and aspatial constraints on a set of data [14].

Features in the database are differentiated by a unique identifier. The structure of the database is addressed by the *feature schema* (a data dictionary, describing the properties of a feature type, their data types, constraints on the data types, etc.). The feature schema is provided by the server using the *DescribeFeatureType* requestand it must be consistent with the OGC feature mode, defined in the OGC Abstract Specification ([53]). The knowledge of the feature schema, used by the server, lets the clients construct features (e.g. for inclusion into the database) that do not raise structural conflicts and integrate seamlessly to the datastore. The standard provides means for specifying specific vendor specific extensions (such as the use of specific Structured Query Language (SQL) constructs on RDBMSs).

The WFS protocol supports the *cascading* of servers, providing opaque access to several datastores for the clients. Cascading servers are capable of performing *homogeneous requests* on other servers (all objects of one query are from one datastore). A Web Feature Server providing *distributed service* also supports *heterogeneous requests*, where the objects of one query may be physically located in different databases (servers). Such services require advanced locking techniques (two-phase commit) from the servers.

The Web Feature Server standard defines the following interfaces (requests):

GetCapabilities This request describes the capabilities of the Web Feature Server in a specific XML document that is modelled closely after the WMS capabilities specification. The capabilities document contain sections about the service metadata—service provider contact information, access URLs for requests, relevant keywords, vendor-specific extensions, feature types with operations defined on them, and other information.

DescribeFeatureType The purpose of this interface is to provide the means for requesting a definition of any feature type that a particular WFS can serve. The feature schema (data dictionary) is expressed in the *XML Schema* ([62]) language[13]. The feature schema should adequately describe all objects that are accepted by that server.

GetFeature The *GetFeature* interface allows clients to retrieve features from a *Simple Feature* datastore. One request can contain multiple queries, selecting features by their properties (attributes). Client can also instruct the server to lock the features that are returned as a result of these queries.

The server responds to this request by sending out data in XML format.

LockFeature The *LockFeature* request is used by *Transaction WFS* for locking specific objects in the datastore. The need for explicit locking results from the inherent statelessness of web connections which would result in unpleasant side-effects when several clients are using the same object for write access (update, delete) simultaneously. Without locking the results of some of these operations would be lost and the database could be corrupted. Using the *LockFeature* request grants an exclusive lock ('ownership') for given objects to the client performing this operation. Other transactions trying to modify the locked object would fail during the lifespan of the lock. Clients can specify the time-out before the server expires the lock automatically (to avoid deadlocks), the completion of the *Transaction* operation on the locked object by the owner of the lock will also release the lock.

The *LockFeature* request can operate on multiple objects, using the filtering system [14]. The implementation of the locking operation is opaque to the client.

Transaction The *Transaction* interface is used to describe data transformation operations that are to be applied to a web accessible datastore. The WFS server translates the client request into the language of the target datastore and executes the transaction. A notification is sent to the WFS client about the result of the operation. The identifiers of the newly created objects are also returned in case of insertion.

The *Transaction* request can perform *Insert*, *Update*, and *Delete* operations which are performed on the objects previously locked by the *LockFeature* interface. *Transaction* can operate on a subset of locked features, with the possibility of leaving others in the *locked* status. The GML data describing the objects to be modified should be in accordance to the feature schema.

3.3 Conclusion

Two methods for exchanging data were discussed in the current chapter—files and services. The specifications of several instances of both methods were covered.

A question rises now, whether to use files or set up a server for exchanging data for a particular project? The choice should be made individually for each case as it depends on many factors. Some points to consider are shown in Table 3.2. Although files are traditionally used for importing and exporting data from applications, services are gaining importance

[13] WFS vendors can also support other schema description languages, such as Document Type Definition (DTD).

Quality	Files	Services
Network connection	not required	mandatory
Latest data	probably	always
Data discovery	complicated	easy
Integration with other (non-GIS) systems	harder	easier
Implementation of access restrictions	harder	easier
Complexity of implementation	lower	higher
Usability in legacy systems	better	worse
Data provider and consumer both active	no	yes

Table 3.2: Comparison of file and service based interoperability methods.

as data providers. Nowadays the geospatial services should be considered seriously for new systems, especially for applications that operate in a Distributed Computing Platform (e.g. the Internet) where there exist plenty of such servers.

If the file based data access method was chosen then the question of file formats needs to be solved. Table 3.1 on page 23 summarizes the properties the file formats that were covered earlier in this chapter. Pros and cons may be different for various situations, but attention should be paid to metadata: skipping it is easy if it is not needed but a dataset is often unusable due to the lack of metadata. Therefore the appropriate support for metadata should have high priority whenever a file format is selected. Topology should also considered essential for large and complicated datasets. From the covered file formats DIGEST is probably most appropriate for complex data while GML may be preferable for data with simpler structure and lower volume. GML should also considered for cases when XML based formats and technologies are used for other (aspatial) data, as the unified processing of spatial and aspatial data can provide substantial simplification for the whole application or technological chain.

When services are found superior to file exchange then the question about selecting an appropriate standard rises. In current situation I would suggest to use relevant OGC specifications, as they provide the best solutions, especially for web services.

Although some proprietary service protocols may appear more powerful and elegant they are often inappropriate for actual use, either for lack of accompanying metadata, insufficient abstraction, or other reasons. Currently available geospatial services are mostly map servers that provide raster images, generated from spatial databases. OGC has specification for similar functionality (WMS) but also for many other services.

The model of services as proposed by the Open GIS Consortium is qualified for a large range of problems, from the very simple ones (e.g. elementary public map servers) to highly complicated, multi-tier data stores that can accommodate several types of data and provide appropriate interfaces for accessing the database (e.g. WFS for vector features, Web Coverage Server for satellite imagery, and so on). The strength of the OGC specifications lies in the modular structure—all services are perfectly usable separately, yet the true elegance of the standards is revealed when two or more services are used to complement each other.

The technical excellence (extensive use of XML technology, good support for metadata, etc.) of the OGC specifications should provide enough motivation for using them instead of proprietary protocols in most (if not all) cases[14]. Open protocols should be supported as only formal standards allow for the most people to get the most benefit.

[14]For products which 'talk' multiple protocols, proprietary and open, like ESRI ArcIMS, the open interfaces should be definitely enabled.

Chapter 4

WMS Servers

An overview of some tools for providing interoperability in the GIS world was given in the previous chapter. The topics of the next three chapters concentrate to the implementations of OpenGIS Web Map Service (WMS) specification, providing information about the servers, clients, and the overall process of solving the real word problems using the WMS.

This chapter analyzes three distinct WMS server implementations. Two map servers, *CubeSERV* and *MapServer*, are discussed briefly in appropriate sections. Most attention is given to the F-Open server, a WMS server implementation developed in Estonia. The author of this thesis has contributed substantial parts to the design and implementation of the F-Open server, which let me provide here a deeper insight into the key points of the design of this application. The chapter ends with the comparison of the three servers.

4.1 MapServer

MapServer[1] is an on open source development environment for building spatially enabled Internet applications. The version of the program is currently at 3.5 [35].

The MapServer was originally developed by the University of Minnesota (USA) For-Net project in co-operation with National Aeronautics and Space Administration (NASA) and the Minnesota Department of Natural Resources. Additional enhancements have been funded by several organizations and private companies.

It is a Common Gateway Interface (CGI) application that can be compiled to run on Linux, Unix and Microsoft Windows systems. The output of the program (a web page) is generated from template, providing easily configurable look. The outputted page can contain the generated map image, controls for zooming, panning and querying map features, and other elements, provided by the owner of the server.

MapServer uses several other open source or free software packages for accomplishing the task of creating spatial applications: Proj.4[2] cartographic projections library, GD library for dynamic creation of images, and many others.

MapServer can load and process spatial data in lots of formats, both vector and raster: ESRI shapefiles and ESRI ArcSDE vector formats, TIFF/GeoTIFF, GIF, PNG, JPEG, and other raster formats. Support for these data formats is in the core of MapServer, many more kinds of data can be utilized by configuring the MapServer to use the Geospatial

[1] http://mapserver.gis.umn.edu/
[2] http://www.remotesensing.org/proj/

Data Abstraction Library (GDAL)[3] for raster and OGR Simple Features Library[4] for vector formats (the development of these two libraries is headed by Frank Warmerdam).

The data may be given in any of the vast number of map projections provided by the Proj.4 library. MapServer can be set up to handle on-the-fly re-projection of the data by the user request.

The output map image is generated by the GD library, the format of the map raster image is any of the image types supported by the library.

The visualization of spatial data is carried out by an user-defined symbology. Map features can be presented by markers (point symbols), lines, or shadesets (area styles). A particular symbolization is bound to an object by a set of rules, described by regular expressions that are combined with simple logic operators (conjunction, disjunction and comparison), on feature's attributes. The rules can also specify the minimum and maximum scales where the map object is visible. MapServer can automatically create appropriate legend for the visualized data, showing the meaning of different graphical elements.

The MapServer system supports *MapScript* which allows popular scripting languages such as Python, PHP Hypertext Preprocessor (PHP) and others to access the MapServer Application Programming Interface. Using MapScript it is easy to create web pages with maps in everybody's preferred language. MapScript gives the users full control over the MapServer program—graphical rendering of spatial data from multiple data sources, map queries, automated generation of the legend and scalebar, and so on.

MapServer understands OGC WMS versions 1.0.0, 1.0.7, and 1.1.0. It can also handle the cascading of other WMS servers. In fact, there is no WMS client in the MapServer distribution, users who want to access WMS datasets are supposed to set up their own MapServer installation that represent WMS queries in the MapServer CGI interface. In this case MapServer is acting as a cascading WMS server whose last leg (closest to the user) speaks its own protocol, not WMS.

Most of the characteristics of MapServer, described earlier, are also available via the WMS interface, the only differences are in the request parameters and server responses (A WMS server sends an map image in response to the *GetMap* request, while the MapServer's native behaviour is to output the whole HTML page.).

MapServer understands the *GetCapabilities*, *GetMap*, and *GetFeatureInfo* requests. The XML response for the *GetCapabilities* request is generated automatically from the MapServer configuration file.

GetMap can return maps in several raster formats, provided by the GD library (PNG, JPEG, or Wireless Bitmap (WBMP)). The result of the *GetFeatureInfo* query can be requested by the client as plain text, HTML formatted by a template, or as GML.

4.2 CubeSERV

CubeSERV is a WMS server that is developed by CubeWerx, Inc.[5], located in Quebec, Canada.

CubeSERV is compliant to the latest WMS standard (version 1.1.1 at the moment of this writing). It supports cascading the services of other WMS servers and all three request types that are specified in the WMS standard. The server supports data in multiple co-ordinate systems and several raster image formats (GIF, PNG, and JPEG) for the *GetMap* request.

[3] http://www.remotesensing.org/gdal/index.html
[4] http://gdal.velocet.ca/projects/opengis/
[5] http://www.cubewerx.com/

It can serve dynamic (live) maps from large (and possibly distributed) databases, being is powered by the CubeSTOR spatial warehouse system.

CubeSTOR is a *geospatial warehouse* system that provides the storage, management, and access to very large volumes of spatial data. Another purpose of the geospatial warehouse is to ensure the uniform data model across the system and keep the appropriate metadata, catalogues, etc. up to date.

CubeSTOR is based on the Oracle RDBMS system. The system includes tools for creating and maintaining geospatial warehouse, *middle-tier* application server, and the presentation interface (the CubeVIEW application). CubeSTOR is supported on multiple computing platforms: Windows NT, Linux and several incarnations of Unix [8].

The CubeSERV WMS server license costs $8000, additional payments are necessary for support and maintenance contracts. The cost of the license of the CubeSTOR Geospatial Warehouse is $27000 (this includes the license of CubeSERV).

The publicly accessible demonstration of CubeSERV (which is a great source of interesting map data due to its large number of map layers and cascaded servers) is located at `http://www.cubewerx.com/demo/cubeserv/cubeserv.cgi`. The CubeVIEW program, available at `http://www.cubewerx.com/wmt/cubeview/cubeview.cgi` can be used as a graphical user interface for browsing the data.

4.3 F-Open Server

F-Open server is a WMS server produced in the *R-Süsteemid* company, capable of servicing the WMS *GetMap* request (it supports several raster image formats for generated maps) and also able to form GML output for the *GetFeatureInfo* request.

F-Open server is a commercial grade WMS server that is known to work flawlessly with wide range of WMS client programs. It accepts requests in WMS protocol versions 1.0.0 and 1.1.0 (the latter is preferred, 'native' version), supporting all three request types that are specified in the Web Map Service standard. A descriptive error message is sent to the client every time the server is not capable of fulfilling a particular request. Client can specify the preferred format of error messages (Three formats are available: an XML document, conforming to the standard exception DTD from OGC, a blank image, or an image with the textual description of the problem.) and image formats in its requests to the server.

The PNG and JPEG raster formats are currently supported for image responses (generated maps and exception indicators) while XML is used for textual data returned from the WMS server to the user. F-Open server can output several types of XML documents—WMS capability definitions, GML data in response to the *GetFeatureInfo* request, and error messages.

The WMS standard specifies the division of data, provided by a map server, into *layers*. In case of F-Open server a layer contains the data one or more feature classes from a Frege map file. It is not possible to use the data from multiple files on one layer, although several layers can display data from one file.

F-Open server does not limit the number nor size of data sources (map files) for a server instance, it is limited only by hardware (mostly by memory capacity). Each layer may have an arbitrary number of associated *styles* that provide different visual looks (graphical languages) of the map.

F-Open server is commercial software that is licensed for a fee. The price is negotiated separately for each customer, depending on the desired functionality (and possible modifications), the amount of data being served, and the number of concurrent users. F-Open server is also offered by *R-Süsteemid* as a part of larger packages, which include the WMS server and customized client software, data file converters, etc.

A working demonstration of F-Open server is located at `http://www.r-systems.ee/F-Open-server/server.php`.

4.3.1 History

The history of the project goes back to the beginning of the year 1999 when the *R-Süsteemid* company decided to implement an Internet map server based on the Frege map engine. The first prototype was implemented as a CGI program written in the *C* programming language. The program started, loaded a map from hard disk, drew the map requested image and exited. This was clearly inefficient as it caused many unwanted activities, such as the initialization of map engine or loading the maps from files to memory. In addition the map visualizing was performed using the drawing engine of X Window System, Version 11 (X11), then converted to X Pixmap (XPM) file which was displayed in user's browser, as the Frege library could do only X11 drawing those days.

Later that year a client-server model was implemented, consisting of a server (mapserver) which handled the map loading and drawing parts, and a client, which was a CGI program brokering user requests to server and presenting a nicely formatted HTML page with the returned map image to the user. Both programs were written in the *C* language, they communicated using a network socket. The server was implemented as a Unix daemon process which loaded the maps during its initialization phase and fulfilled subsequent requests by rasterizing the vector maps into images.

The CGI part parsed a template HTML file and when it encountered an item from a set of proprietary macros inserted into that file it expanded the macro into the appropriate control (communicating with the server when necessary), resulting in a standard HTML page which was delivered to client browser. These macros included a set of selectable buttons for map zooming, a map picture itself, map panning controls and more. This way the appearance of the map pages could be easily re-designed (by a graphical designer) without the need of any re-programming.

Around the same time support for several other output formats were added to Frege using the free *GD* library[6], developed by Thomas Boutell. The GD library (and therefore also Frege) supports PNG, JPEG and WBMP images.

Later, multi-threading support was added to mapserver and it became tmapserver. Multi-threading made the server usable in real-world situations where several requests could arrive at the same time, that is another request might arrive at the time server is still processing the first one. Previous versions produced correct output only when the request were sent sequentially (requests could not be overlapping in time). tmapserver was capable of servicing ten simultaneous connections.

Other improvements included the rewriting of the client module in the PHP programming language, partly as an exercise to learn new technologies and partly for making the process of changing the mapserver front-end easier. The overall architecture (client-parsed macros, etc.) remained the same.

In the spring of 2001 tmapserver was redesigned to meet the requirements of the OpenGIS standards, the resulting product was renamed to F-Open server.

[6] http://www.boutell.com/gd/

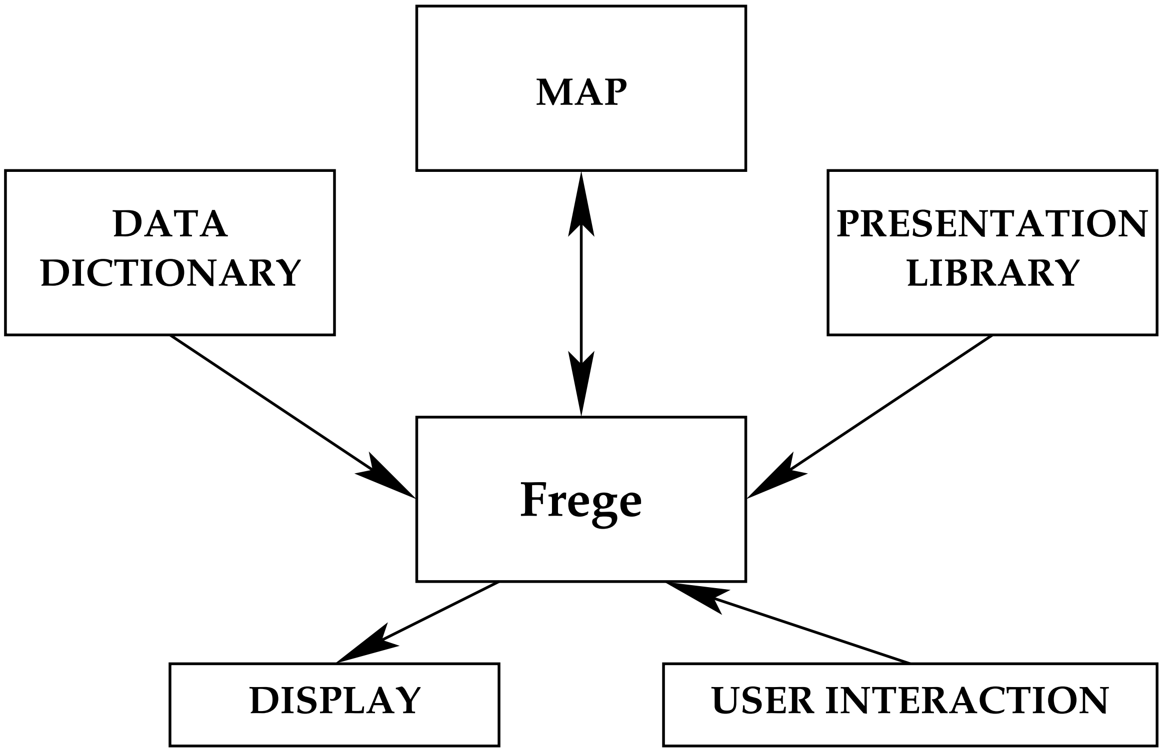


Figure 4.1: Handling maps with Frege.

4.3.2 Frege

Frege[7] is a library of functions for handling digital vector maps. Frege is a direct descendant of the Mapper library, described in my diploma project [28]. The author of this work is one of the principal developers of the map handling library. The Frege library is grown out from the program code that displayed nautical charts, now it is completely reworked and its internal structure can accommodate the majority of different spatial data formats that are in use nowadays. The other strong point besides universality that characterizes our map engine is performance—the first application that used Frege was a hydrographic data acquisition software on board of a survey ship where fast map drawing routines were necessary, as the computers would not have spare processor cycles for collecting data from several sensors [47].

Frege includes subroutines for loading and saving maps, displaying the maps with arbitrary visible area and scale, changing the map content (editing of map objects or metadata), and making spatial queries on maps. The principal schema of operation is shown in Figure 4.1.

The inner data model for Frege is derived from the S-57 standard. In addition the ability to use different kinds of maps (based on different data dictionaries) with different presentation libraries is added by *R-Süsteemid*. Maps can use any of the four topology models, defined in S-57 (spaghetti, chain-node, planar graph, or full topology), all map editing functions are aware of topology. A rich set of meta-information is also attached to each dataset, describing the co-ordinate system, dataset parameters (e.g. creation time and creator), number of objects, etc.

[7] The name 'Frege' was chosen after the logician G. Frege, to stress the importance of semantics in communication (The Frege's triangle links together *sense*, *expression* and *referent*.).

Maps are stored in a flat files for efficient loading times. All file operations in Frege are done with these files, data can be imported to Frege format via converters for S-57, *dgn*, shapefiles, and MapInfo *tab* and *mif/mid* formats. Export of modified map data is possible to S-57 files.

Frege contains no hard-coded information about the map semantics, it is stored in *data dictionaries* instead. Each map is associated with a data dictionary that provides the definitions for all classes of feature objects and a possible set of attributes for each class. With different data dictionaries Frege is applicable in very different application domains.

Neither data files nor data dictionaries contain no information about the visual style of map elements. Separate *presentation libraries* are used for visualizing map content. There may be more than one presentation library suitable for one data dictionary (a type of maps), making the alternate visual representations of data very easy, one just need to draw a map with another presentation library.

Presentation library is a powerful concept for visualizing spatial data. It is a graphical language that provides look-up tables for matching each map object (feature) to a set of graphical primitives, based on object's class, geometric primitive and attribute values during the map drawing process (see Figure 4.2). A presentation library can possibly contain several object look-up tables, color tables, and other parameters, which can be selected individually. These parameters let the end user (e.g. WMS clients) select among the alternative visual looks for the same set of data, choosing the most characteristic and expressive form, based on user's needs.

There are several types of graphical primitives:

- line drawing styles (simple ones (*LS*) for variable width coloured lines and complex line-styles (*LC*), where line is visualized by a repetitive pattern).
- area fill instructions, either with one color (*AC*, four levels of transparency) or with repetition of pattern symbols (*AP*).
- symbols for point locations—*SY*.
- text drawing instructions with selectable font family, size, and justification (*TX, TE*).
- complex symbolization primitives (*CS*) provide hooks for external functions (from a dynamically loadable library) that can provide more complicated symbolizations than the look-up tables of the presentation library. Complex symbolization procedures can also take advantage on the user-changeable presentation parameters for providing more dynamic visualization.

All symbol and pattern definitions are given in a special vector drawing language which is independent of the final output system.

Two target systems are available for map drawing—X11 (network transparent windowing system, used mainly on Unix workstations) and the GD library, the latter providing support for drawing into various raster image file formats. Adding new code for other target systems is straightforward due to modular design of the output subsystem of Frege.

A work is currently underway for interfacing Frege with a DBMS. PostGIS[8] will be used for a spatial database, feature attribute data would be located in any Open Database Connectivity (ODBC) compatible DBMS. These arrangements take advantage of the powerful engines of today's database systems for using them for attribute (aspatial) queries using the SQL. The Frege-database link is used for the WMS server of the registry of assets of RMK.

[8]PostGIS (`http://postgis.refractions.net/`) is a set of facilities built on top of a PostgreSQL relational database to achieve the conformance with the OGC *Simple Features for SQL* specification.

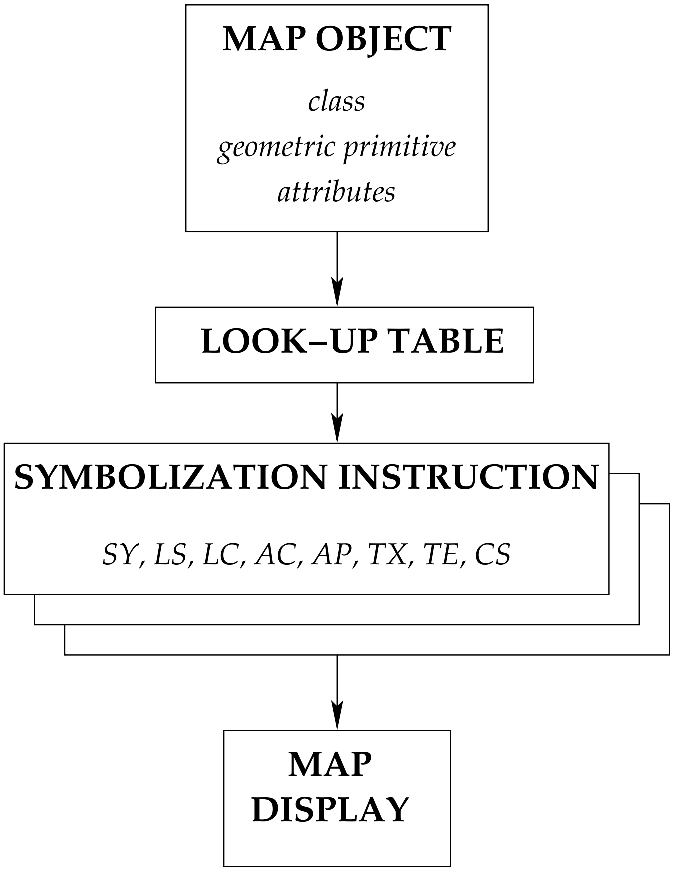


Figure 4.2: The work-flow in the Frege presentation library.

Depending on the results of the current project a continuation project may be following for allowing also spatial queries to be carried out by the database engine on behalf of Frege. Using the PostGIS database for spatial data storage also makes it easier to co-operate with other systems that use the *Simple Features for SQL* ([42]) standard for accessing geodata.

4.3.3 System Architecture

F-Open server is composed of two parts, *front-end system* and *server proper*, as shown in Figure 4.3 on the next page. These two parts are separate programs that communicate over the network. Such design gives great flexibility in the configuration, as the server proper and front-end may run on separate computers. Such separation have positive effects on load balancing and the overall security of the system. Indeed, only the front-end (with its relatively small code base) is required to be on the 'outer' side of the firewall, the server proper may be located behind the firewall. All network ports other than the port used for communication with front-end, may be left closed on the computer that hosts the server proper.

One important aspect of the F-Open server is the *image cache*. It is a structure that stores all generated map images together with the parameters (image size, location, map layers, etc.) of the request that are needed for image generation. A quick look-up is performed from the cache for every *GetMap* request that arrives, and if the parameters of the current request match a cache entry the time-consuming rasterizing phase is bypassed and the map image from the cache is returned to the client. The image cache has a finite size, if the cache is full and a new request arrives, the least recently used entry is purged from the cache and

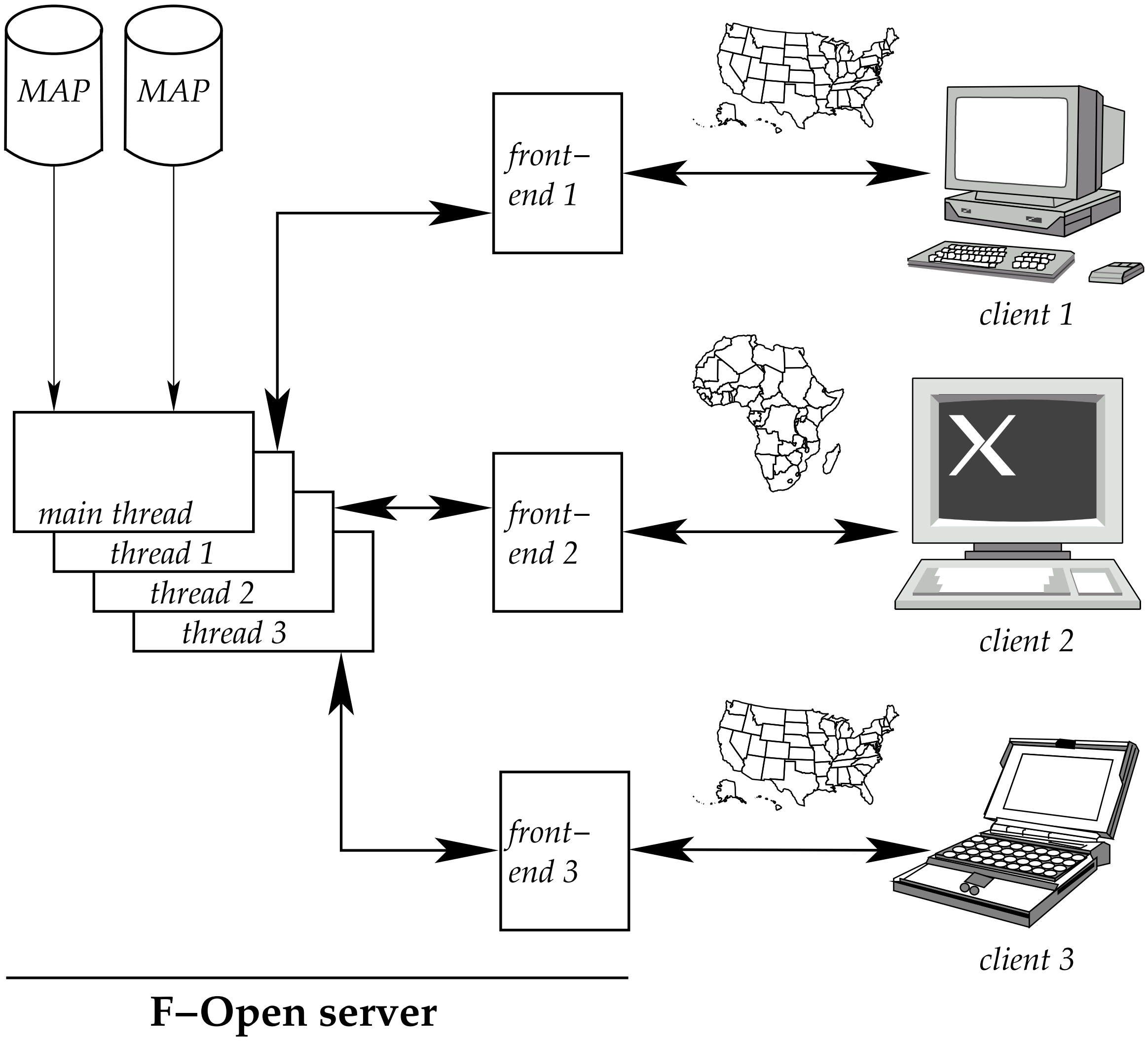


Figure 4.3: The schematic diagram of the F-Open server.

the result of the new request is stored in its place. The size of the cache is adjustable from the configuration file.

F-Open server may be instructed to answer to the *GetFeatureInfo* requests for certain WMS layers. When the server gets such query it searches through the provided layers and returns the information about all objects that contain the point of interest, as specified by the client. The whole set of spatial (co-ordinates) and aspatial (attributes) information that is known to the server about these objects is returned. The information is supplemented with the descriptions from the data dictionary (feature class name, attribute names and values, etc.). The feature data is returned in the GML format, the owners of the server can specify proper XML schemas and namespaces for each queryable layer.

According to the OpenGIS WMS standard an *exception* is raised and an error message is sent back to the client every time something unusual happens. A client can request exceptions to be reported as a standard XML notation document, as a blank image, or as an image with the textual description of the message.

Front-end

The task of the F-Open server front-end system is to do the initial processing of the parameters of the request and forward them to the server proper and to send the results(images or XML documents) back to the user after the server has completed the processing of the user request.

The front-end is written in the PHP scripting language and is executed by the Apache[9] web server (the interpreter for the PHP language is compiled into the web server). The web server's job is to listen to the HTTP requests and start the F-Open server front-end script when the HTTP request comes to the address of F-Open server (see also Figure 3.2 on page 29).

The F-Open server front-end then starts to process the WMS request by parsing the parameters of user inquiry. Besides the formal syntax checking it also does away with the syntactic differences between versions of the WMS protocol when necessary. For example, WMS version 1.0.0 defines the accepted values for the `REQUEST` parameter as `map`, `capabilities`, and `feature_info`, while the WMS version 1.1.0 specifies the values `GetMap`, `GetCapabilities`, and `GetFeatureInfo`. This step is needed to keep the communication protocol between front-end and server proper the same, regardless of the WMS protocol version specified in the client's request.

Front-end is also responsible for version negotiation with the client, as specified in the WMS standard [59]. F-Open server is capable of servicing requests for WMS versions 1.0.0 and 1.1.0.

Server Proper

The server (*server proper*) is the part of F-Open server that is responsible for fulfilling the requests of WMS clients. It interacts only with the F-Open server front-end, not with the WMS clients.

The server proper is a multi-threaded program (daemon) that waits for requests from the users (via the front-end system) and spawns a new processing thread for each user connection (Figure 4.3). All these program threads run in parallel and access the same data store. No signaling between threads is necessary as the WMS server does not modify the data (some data structures, such as the image cache, are explicitly locked when write access is mandated).

The F-Open server proper is implemented in the *C* programming language, it uses the *libxml2* [10] library (created by Daniel Veillard) for parsing and validating XML documents.

F-Open server has a configuration file where the server administrator can specify the exact behaviour of the server instance. The file is an XML document that is basically identical to the WMS capabilities XML. Some additional elements are introduced for F-Open server-specific components, such as the names of log file, map files, presentation libraries and parameters, and so on. The configuration file is loaded at the start and validated for correctness against the DTD. This step eliminates the need to check the presence of all mandatory parameters and the overall well-formedness of the configuration in server code, removing much hassle. One can also force the server to reload its configuration by sending the `HUP` signal to F-Open server process (`killall -HUP fopenserver`).

The F-Open server writes traces of all events to the log file. Actions that trigger logging include normal user requests (client host name, request parameters, etc.), error conditions and other relevant information. Each record in log file is written together with a time-stamp

[9] http://www.apache.org/
[10] http://www.xmlsoft.org/

and thread identifier, providing additional information for later analysis on server's usage patterns and/or tracing the misbehaving of the program.

4.3.4 Future Plans

Plans for the future enhancements for F-Open server include the implementation of WFS functionality, support for cascading servers, and the ability of dynamic (on the fly) re-projection of maps. High priority is given for developing more data import converters for the Frege library.

An important addition for the map server would be the ability to 'speak' in different languages. This improvement would create the possibility for the user of the map server to choose the language (via client applications) for the dialog with F-Open server, allowing the internationalization of WMS systems—e.g. the public server of Estonian Maritime Administration should be able to communicate in Estonian, English, Finnish, Russian, and Swedish languages (Most people interested in Estonian nautical charts are using these languages.).

The texts that server sends to a client, meant for human users (layer names, descriptions, error messages, etc.) will be in the language that the user selected in his request. Available languages are defined in an extended capabilities XML document.

This idea will be realized as an extension to the WMS protocol, as OGC has not yet created a specification for multi-language support in its services. The language selecting option would thus not be available in all WMS client programs.

4.4 Conclusion

This chapter gave an overview of three distinct WMS servers: MapServer, CubeSERV and F-Open server . A more detailed description of the design and implementation F-Open server provided some ideas about the techniques used in implementing the OGC WMS specification.

Following list shows some points of comparison between the three WMS servers introduced in this chapter. These arguments are applied here to compare F-Open server, MapServer, and CubeSERV, but they can be useful for evaluating other map servers as well.

Functionality The most important criterion for anybody selecting a map server would be the *functionality*. All three applications implement the OGC specifications and although each has its small quirks they can all interoperate quite flawlessly. They are all on par in regard of this property.

Supported spatial data formats Section 3.1 demonstrated the great variance in the spatial data formats. The primary component of every GIS system is the data and therefore everyone needs to choose the right tools that can handle the formats of the data that are needed for a particular case.

MapServer can handle both vector and raster data of different kinds, when MapServer is built with the OGR and GDAL libraries it can utilize most widely used data formats.

F-Open server does not support many different data formats and cannot handle raster data at all, but its strength is in the handling of IHO S-57 format that is very important to all mariners (MapServer can also load S-57 datasets but it does not have proper support for correct representation of the data, according to the S-52).

There are few facts about the CubeSERV but I suppose it can also load geospatial data in most common formats.

Output formats Another aspect of comparison is the type of output. It is important for cascading WMS servers and also for end users who may want to save the map images and use them in other programs.

While the format of the *GetCapabilities* response is fixed by the OGC the formats of the responses to the other two requests are not.

F-Open server, MapServer, and CubeSERV all support PNG and JPEG raster image formats the *GetMap* request (CubeSERV can additionally produce GIF and MapServer WBMP images). These image types are the most common in Internet and should pose no problems.

For the *GetFeatureInfo* all three support the GML format, MapServer and CubeSERV can also output features in plain text or HTML format. Again, the GML is the preferred format by the OGC and should become the standard for spatial data exchange in Internet.

Cost The total amount of resources (i.e. time, money, knowledge) are nearly always limited and should be used up most efficiently.

The price of the WMS server sums up from the cost of the WMS server software, other software that is needed (operating system, database systems, additional libraries, etc.) and hardware. These sums are one-time investments (until the upgrade of software or hardware) but there are also running costs associated with map servers. The expenses for actual spatial data should taken into account as well as the cost of network bandwidth, which may be the largest by far of all expenses for a WMS server.

MapServer is a clear winner in this section as it has no associated costs for software.

Performance The overall performance of a map server depends on many factors starting from the hardware, the amount and organization of the data, network bandwidth, and ending with the quality of implementation of the WMS server.

The importance of the performance varies with the purpose of the server, greater performance is never bad but some critical systems will need good performance at any cost.

It is nearly impossible to compare the performance of three different systems that can use various spatial data formats. However, these are some 'rules of thumb' for evaluating the performance of map servers:

- The execution of CGI programs (MapServer) is inherently slower than a separate daemon (F-Open server) due to the additional cost of allocating resources for the new process.
- Most GIS software has a 'native' file format, using data in this format would yield the best performance. E.g. MapServer should give the best performance with shapefiles and F-Open server with S-57 data.
- Generating the map image is the most common operation of the WMS server, it is often also the most time consuming operation. F-Open server uses its image cache to leverage this problem—the ready-made image is taken from the cache if there has been a request with exactly the same set of parameters lately.

Chapter 5

Web Map Service Client Programs

This chapter is dedicated to the client side of the Web Map Service specification. Several client applications are described in this chapter.

The complexity of the client applications ranges from nil (the person inserts the URLs manually into the standard web browser) to great (the WMS client is part of a much more complex application). Also, the applications that provide access to WMS servers often do not implement the whole specification. Some of them do not support the *GetFeatureInfo* interface, some can handle spatial data only in certain co-ordinate systems, some have improper user interface (user cannot choose styles for layers or cannot save the map image, for example), etc.

There are an increasing number of WMS client applications available, anybody who needs to get data from a WMS server should choose an appropriate one for his needs. This chapter shows different ways of implementing client-side applications for the OGC WMS protocol, programs are chosen to reflect the diversity of methods in various implementations of WMS clients.

5.1 CubeVIEW

CubeVIEW (Figure 5.1 on the next page) is a data visualization application for the CubeSTOR data warehouse system from the CubeWerx, Inc. It includes a web-based geodata navigator that can act as a WMS client program (Described as a template application for on-the-fly generation of HTML and JavaScript code for various spatial data visualization frontends). CubeVIEW is a CGI program that runs in the service provider's server, user interaction with the program is done using the elements of HTML forms (text fields, buttons, etc.). CubeVIEW is provided free of charge to the purchasers of CubeSTOR or CubeSERV systems [8].

CubeWerx has created a web demo of the CubeSERV/CubeVIEW system (works with all modern browsers), available from `http://www.cubewerx.com/wmt/cubeview/cubeview.cgi`. The demo site defaults to CubeWerx server but it is possible to specify other WMS servers as well.

CubeVIEW provides easy navigation (centering, zooming and panning), manual selection of scale and center point, image type and size and also other parameters. As it is built without using the Java language nor the capabilities of the most recent web browsers, it does not provide the 'dragging zoombox' functionality. This is nevertheless hardly noticeable when actually using the program, as the navigational part is very well thought-out and intuitive.

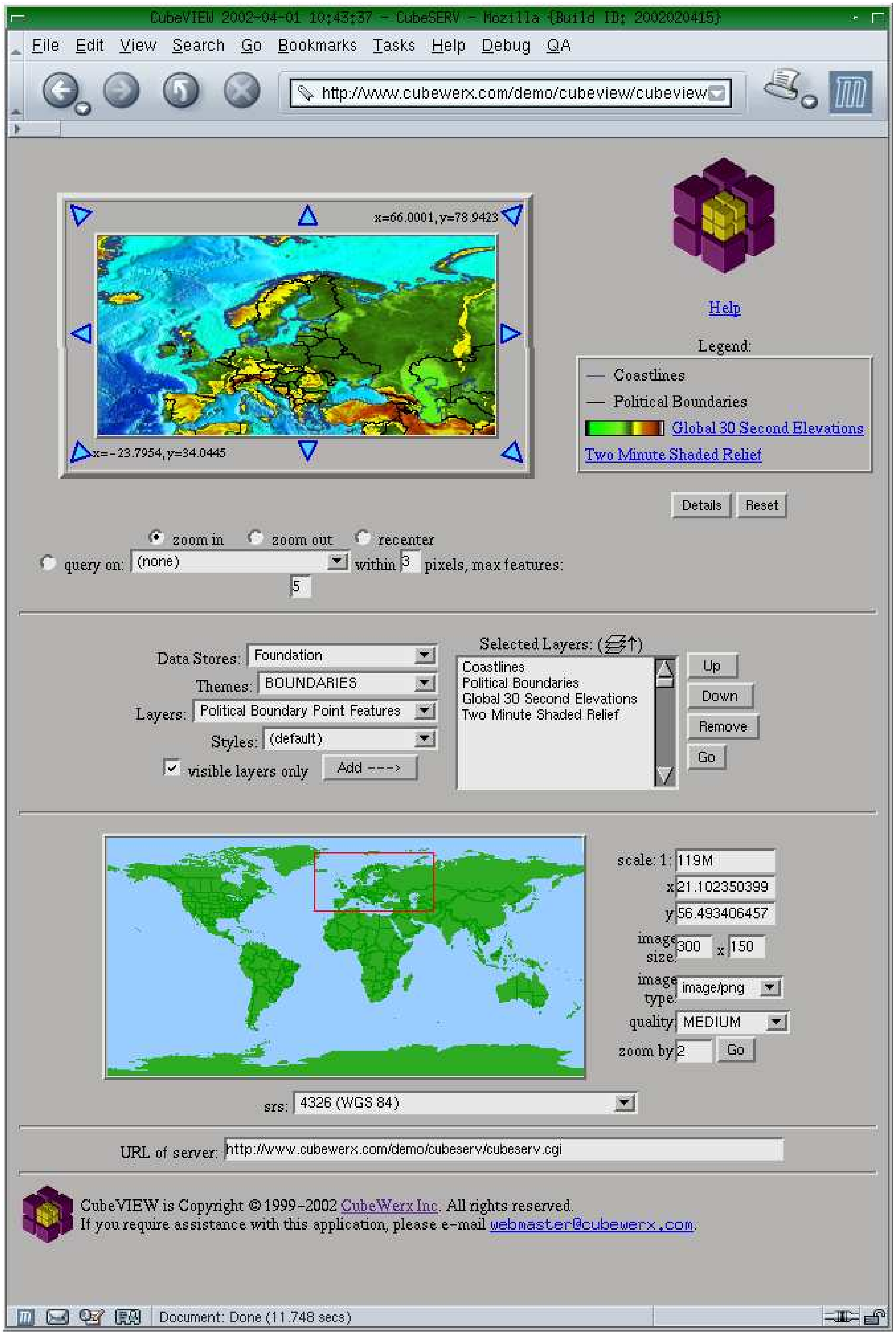


Figure 5.1: CubeVIEW, a WMS client interface from the CubeWerx, Inc.

User can easily add or remove layers from the list of visible maps, using hierarchical selections. Presentation styles can be selected separately for all layers.

The user interface contains many more elements in addition to the WMS map image and navigation controls. The most visible is the world map with a rectangle (or point, depending on scale), showing the area of the main map. There is also legend of visible layers (if supported by server(s)), descriptions of the server and activated layers are available behind a hyperlink.

CubeVIEW supports large number of co-ordinate systems but unfortunately not the L-EST (EPSG:3300) system, widely used in Estonia. Therefore it is not possible to use the CubeVIEW interface for accessing the majority of Estonian geospatial databases.

5.2 WMSClient

WMSClient (Figure 5.2 on the following page), a flexible WMS client application, is a product of the *R-Süsteemid* company. It is implemented in the Java programming language (needs Java version 1.3.1 or higher), which gives platform-independence for practical purposes. Indeed, one can supposedly execute this program on any computing platform where there is an implementation of the Java Virtual Machine (JVM) (there are JVMs for all major operating systems nowadays). WMSClient has been verified to work without modifications on Microsoft Windows 98 and Linux operating systems.

WMSClient can run as a stand-alone program or as an *applet* inside the web browser. In either case it runs in the local machine of the user (as opposed to the CGI approach) which improves the power and quality of available tools and widgets (especially for user interface). On the downside are the higher requirements for the local computer hardware (as compared to the CGI approach again) which can result in slower execution speed.

The program supports all three request types of version 1.1.0 of the WMS standard.

Currently there is no built-in map projecting and datum conversion formulae. for handling different SRSs WMSClient relies entirely on the server to provide bounding boxes for all Spatial Reference Systems in all layers in *GetCapabilities* response—when the server supplies the bounding box co-ordinates in the units of a particular SRS then it is trivial to calculate the co-ordinate parameters of the new request using only elementary arithmetics[1].

It also supports the *composite* (*blended*) map images where the actual map, as shown to the user, is composed of two or more raster images from multiple servers. This feature mimics the behaviour of a cascading WMS server but purely on the client side—there is no need for setting up an intermediate server, the computer user can select for blending any servers he likes. When the user has selected the composite image then all following actions (such as changing the scale) are carried out synchronously for all servers. The creation of the composite image is accomplished by modifying the alpha channel (adding transparency) of the upper images. The images must share a common co-ordinate system for blending.

The user interface of the application is not sophisticated but it offers complete control over the process of selecting desired information layers, navigating, and making feature info queries. Multi-level history is available: users can move back and forth in previously obtained map images without reconnecting to the WMS server. On-line help is provided for quick reference of the user. There is also menu entry for saving the map image in case there is desire to process it further with other tools.

WMSClient is available for downloading at `http://r-systems.ee:7007/WMSSvenska/Applett/`.

[1]The first query is done using the bounding box co-ordinates, subsequent request are based on the user interaction and the parameters of the previous request.

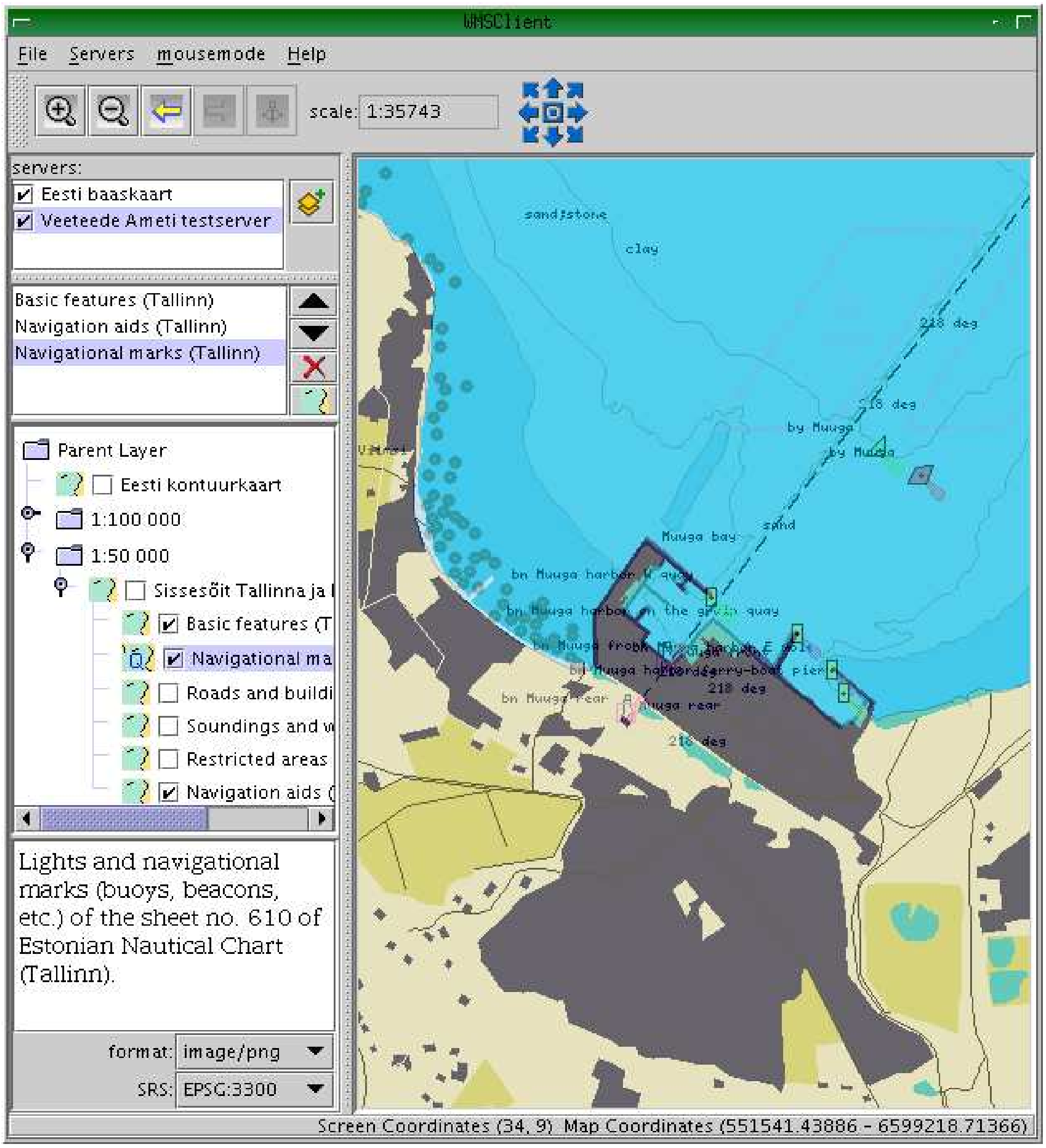


Figure 5.2: WMSClient screenshot. Muuga harbour. The map image is blended from two sources—Estonian Base Map from the WMS server of Estonian Land Board (background) and a nautical chart from the WMS server of Estonian Maritime Administration (foreground).

5.3 WMS Client Toolkit

WMS Client Toolkit is an toolkit to facilitate the production of specialized WMS client applications, developed by *R-Süsteemid*. The toolkit contains several components, written in Javascript and PHP programming languages that can be used for making WMS client applications, tailored for particular needs both in visual appearance and in functionality. The applications would run in the web browser. The *WMS client toolkit* uses technology that is only available in the Microsoft Internet Explorer (version 5.0 or higher) browsers, thus restricting the potential users to people who can (and want to) use this particular breed of browsers.

The PHP component makes HTML pages from the XML and Extensible Stylesheet Language (XSL), using the Sablotron[2] XML toolkit. XML data is fetched from the WMS server (server capabilities XML or GML), necessary XSL files are provided with the toolkit.

Javascript is employed for user interface elements (dynamic HTML): menus, form processing, etc. Javascript program is embedded into the HTML page and is executed in the web browser (in the computer of the user).

The layout of the HTML page that forms the interface to the WMS client is not dictated by the toolkit and is freely customizable by the creators of the client application using HTML and Cascaded Style Sheets. One particular example is shown in Figure 5.3 on the next page.

All request types of WMS protocol version 1.1 are supported. The result of the *GetCapabilites* is stored inside the HTML page that represents the user interface, eliminating the need of requesting capabilities XML from the WMS server for every action. The HTML page, displayed to the user, usually includes the map image and several other user interface elements for setting the query parameters, such as selecting the map layers from a list and choose appropriate presentation style for each layer.

The toolkit provides also additional features for controlling the behaviour of the client program:

- The GML output of a Web Map Service server is translated into HTML and shaped into tabular form for easy perception by humans, the result can be seen in Figure 5.4 on page 52.

- History buffer for storing recently loaded images. The images from the buffer are displayed immediately, without querying the WMS server. User can move back and forward in the history buffer.

- Combination of map images from multiple servers. The blending is done by altering the alpha channel information, thus providing partial transparency of images (except the bottom-most).

- Various navigation controls for panning and zooming the image, 'drag-zoombox' (drag a box on the map and enlarge part of the map inside that box), requesting feature information, setting and returning to anchor point. Text fields and drop-down boxes are provided for directly changing the other parameters of the *GetMap* request (e.g. co-ordinate system and the size of the image).

- Metainformation about the server and layers, extracted from the capabilities XML and provided to the user in a nicely formatted table.

This toolkit cannot perform conversions between co-ordinate systems, for calculating the co-ordinates for the *GetMap* and *GetFeatureInfo* requests it uses the same method as

[2]http://www.gingerall.com/charlie/ga/xml/p_sab.xml

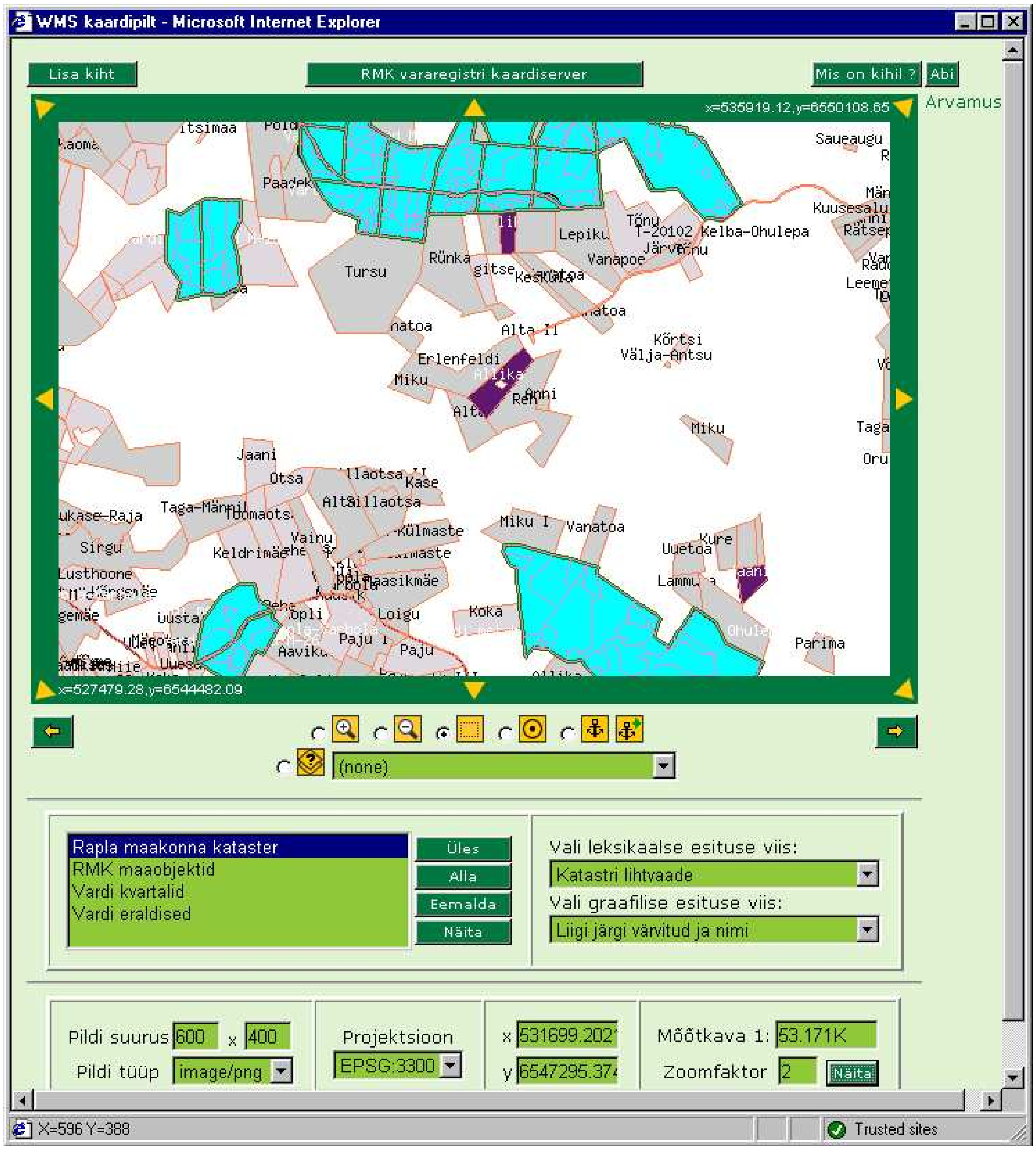


Figure 5.3: An example of a WMS client based on the *WMS Client Toolkit*—the map browser of the registry of assets of RMK.

WMSClient: map co-ordinates are calculated from the `<BoundingBox>` element of the WMS server capabilities. Therefore the clients that are based on this toolkit are usable only with those WMS servers that set the `<BoundingBox>` element correctly for each map layer (The WMS standards does not require the inclusion of this element into the description of a map layer.).

Currently there are two applications built with the WMS client toolkit: EMA public map service client program (Section 6.1) and map browser for the registry of assets of Riigimetsa Majandamise Keskus /State Forest Management Agency/ (RMK) (Section 6.3). The EMA

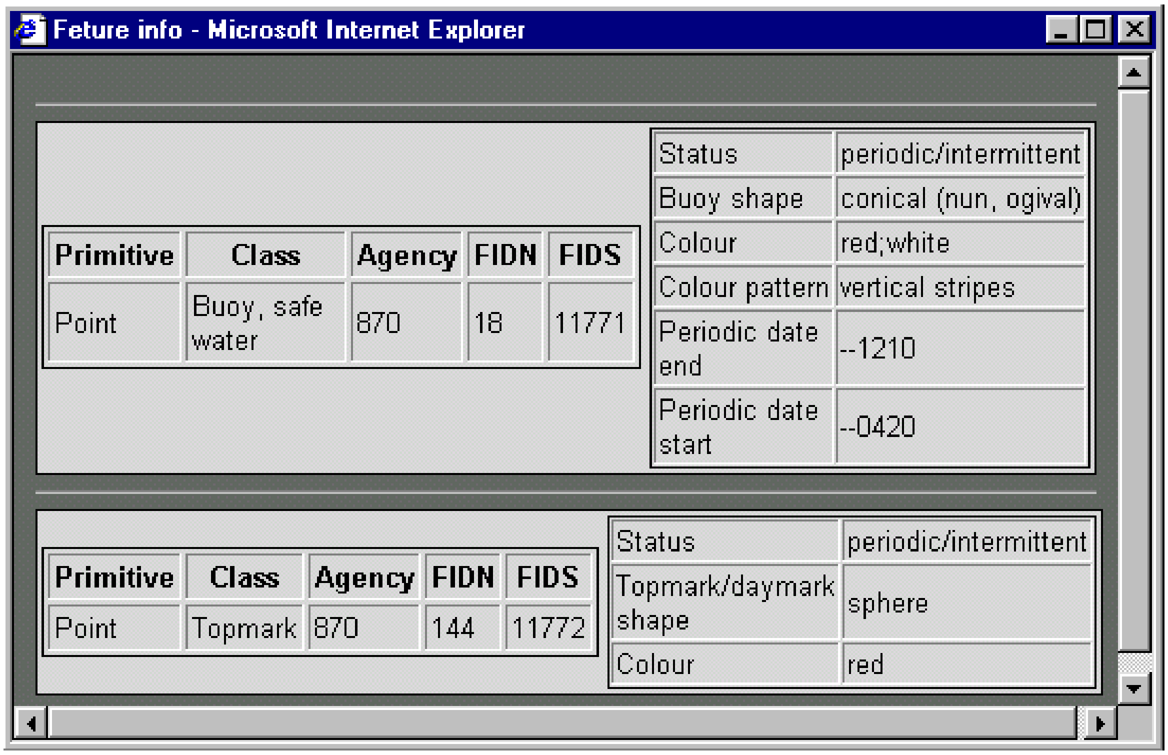


Figure 5.4: An example of the view, displaying the results of the *GetFeatureInfo* request. The figure shows information about two map features, a buoy and a topmark, from the server of the EMA.

public map service client program is located in the EMA server and is accessible at the URL `http://195.50.203.246/wmsc/`. The map browsing client of the registry of assets of RMK is located at `http://r-systems.ee:7007/rmk/WMS/rapla.html`. The user interface of this application is shown in Figure 5.3.

5.4 FRED Map Editor

FRED is a map editor for X11 environment in Unix-like operating systems, developed in *R-Süsteemid* by the author of this paper. FRED is built on top of the Frege map handling library.

The map editor has clean graphical user interface which allows precise positioning and zooming, simple selection of objects by pointing to the map or selecting from list of all objects and quick display of important information about selected map objects. FRED also sports an interactive, hypertext based help system.

The map editor uses Frege data model, which is structured closely after S-57 (see section 3.1.3 on page 18). Supported editing functions include the creation, modification and deletion of feature objects (object's identifier, class name, attributes, relationships with spatial objects), spatial objects (position and attributes, relationships with other spatial objects). Topology is supported during all spatial editing operations.

FRED can simultaneously display unlimited number of vector maps. The order of maps is flexibly adjustable by the user. The maps can be of different types (based on different

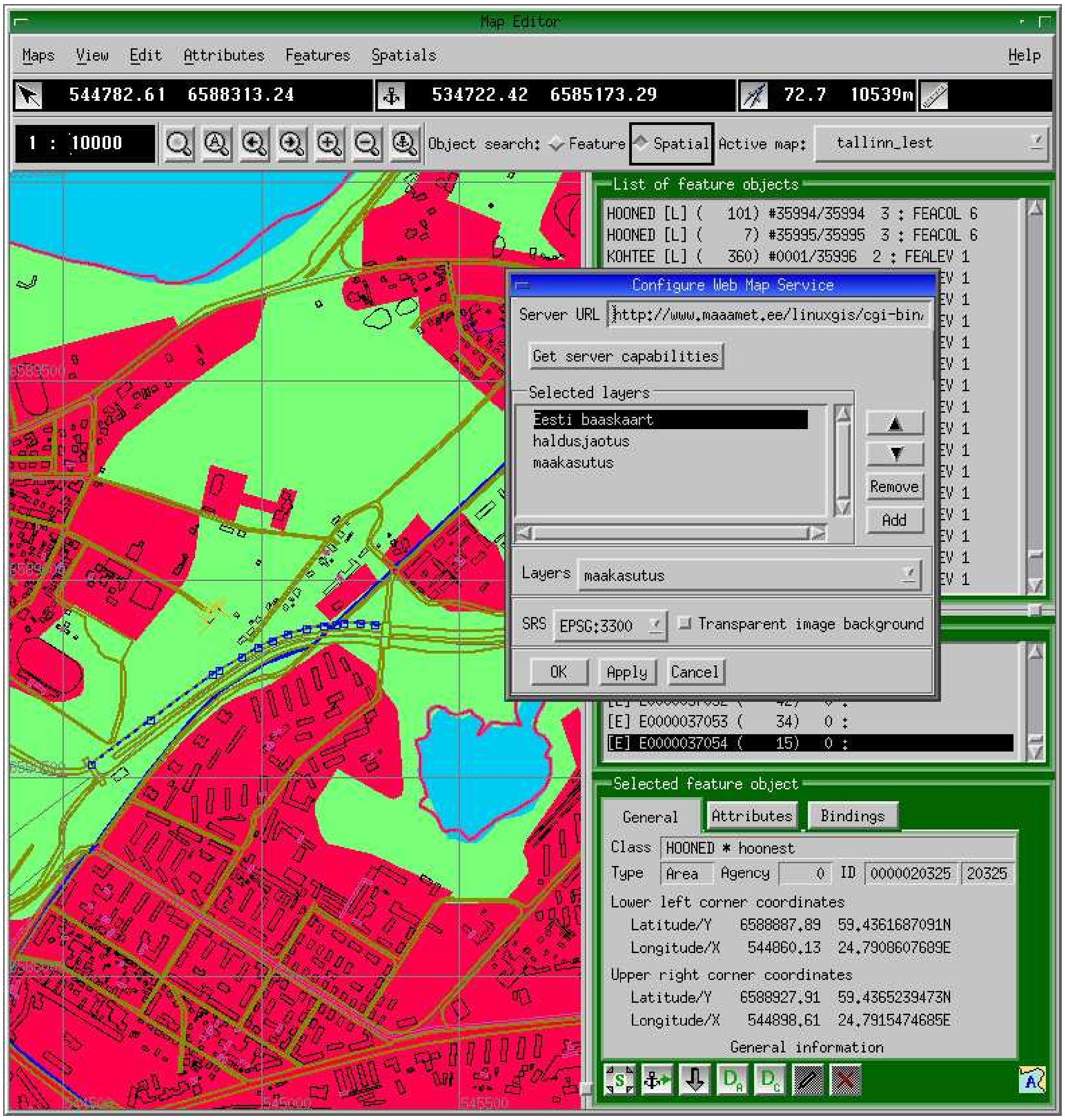


Figure 5.5: Screenshot of the FRED map editor. Lasnamäe district, Tallinn. Street map (line features) is shown on top of raster image (Estonian Base Map, area features selected), fetched from the WMS server of ELB.

data dictionaries)[3], user can choose the visible object classes for each map separately. The appropriate presentation library can be selected at map loading, presentation parameters can be changed later for each map layer. One loaded map is always marked as *active*. User can interact with the active map—select objects either from the list of all objects or directly from the map (rectangular search), or perform other operations. One map can be set *editable*, meaning that the user can modify the contents of that dataset. When no map is marked editable FRED acts as a map viewer.

[3]All maps are stored in Frege datafiles, the process of conversion from and to different file formats is carried out by using external converter programs.

FRED has the ability to load a raster image, acquired from a WMS server, onto the 'bottom' of all vector maps (i.e. vector maps are drawn on top of the image). The image is 'glued' to vector layers and moves automatically when user changes the viewing position or scale (automatic *GetMap* request is sent to the selected WMS server with new parameters). This feature extends the capabilities of the FRED map editor, giving the users of FRED an additional layer of visible background information for positioning or digitizing purposes.

At the moment FRED supports only *GetCapabilities* and *GetMap* queries for the WMS specification version 1.1.0. *GetFeatureInfo* request is currently not performed but is definitely planned for future releases, as well as interaction with WFS servers (editing of remote datasets).

A configuration dialog is provided for selecting WMS servers and changing query parameters (choosing layers and SRS for map queries). User can also choose to temporarily switch off the display of the WMS image, e.g. for closer examination of data on a vector layer.

The screenshot of FRED can be seen in Figure 5.5 on the preceding page.

5.5 Conclusion

This chapter drew some examples of client applications for Open GIS Consortium WMS protocol. Today there are many programs available with different concepts for carrying out the same operations. Everyone can choose the right tool according the purpose, like, and technical restrictions. The browser-based clients are great for public services—no additional software is required (the existence of web browser in every computer is a norm nowadays) and their user interface often looks familiar (when the application is using only standard-looking controls, like CubeVIEW). Other, more specialized WMS clients, like FRED, are mostly intended for specialists, who want to integrate the output of a Web Map Service server with other data in a professional environment. The downside of such tools is the increased complexity, resulting in higher learning curve.

The user interface of HTML-based clients (like CubeVIEW and the clients that use WMS Client Toolkit) is restricted to small set of elements, defined in the HTML standard. While using other programming environments (Java, Microsoft Windows, X11) give more widgets and amusements to the hands of an application developer it also restricts the potential clients to those who can and want to use such programs on their favourite platforms.

The basic functionality is the same for all programs studied here, they all can request the capabilities XML of the Web Map Service server, present the information to the user, and display the map images, requested from the server with user-specified parameters. All clients except FRED support also the *GetFeatureInfo* interface for providing more information about the map content. WMSClient and *WMS Client Toolkit* do not contain program code for co-ordinate transformations and rely on the server for providing accurate information about the map's co-ordinates.

I will reiterate the strong points of the covered applications:

CubeVIEW has support for highly structured data sources (WMS servers), it also provides an indication of the area, visualized in the main image, on the world map which is useful when one works with mid- and small-scaled maps. CubeVIEW support can handle data in many co-ordinate systems.

WMSClient is a Java program that can run inside web browser (if it supports Java) or as stand-alone program on various platforms. The most innovative solution is the possibility to blend the images from multiple WMS servers into one map that is presented to the user. Such operation has similar effects as the server cascades but it is implemented completely on client side, thus eliminating the need of setting up an WMS server (or troubling the server owners). It has also an image cache.

WMS Client Toolkit provides the blending of images from multiple servers, supports *GetFeatureInfo* request and keeps the cache of map images for fast reloading of previously acquired images. It is a set of tools for creating applications with various functionality and appearance, not a ready-made program.

FRED is a stand-alone program, of which the WMS client is only a tiny part, providing the background information. The implementation details differ from other applications covered here but the user interface and overall work-flow of the WMS part are similar to CubeVIEW and other browser-based clients. It does not currently support *GetFeatureInfo* request nor the selection of styles for map layers but can perform co-ordinate transformations between some co-ordinate systems.

The main conclusion drawn from this chapter is the great variance in the implementations of WMS client programs, everyone should be able to choose the right tool for his necessities.

Chapter 6

WMS in Action

The focus of this chapter is on the projects that use the WMS standard for solving the problems that are rising from the real necessities of the GIS users.

I will introduce three systems that are under development in the *R-Süsteemid* company. Although there are many similar projects done around the world these undertakings are among the first to use OGC specifications in Estonia. All these systems are currently (spring 2002) in prototyping or specification stage, hopefully they will be completed soon and become the criteria for other similar projects.

6.1 Public Map Service of the Estonian Maritime Administration

Estonian Maritime Administration (EMA)[1] is the sole producer of nautical charts in Estonia. The purpose of this project is to provide an overview of the maps produced by the EMA and also provide background maps for other services. The output of the map server is not qualified for navigation on the sea.

There is a working prototype of the system which contains a couple of chart sheets, available at `http://195.50.203.246/wmsc/` (the WMS server is located at `http://195.50.203.246/F-Open-server/server.php`). The final installment and testing of the public map server is presently bogged due to the lack of funding from the EMA but should be completed at the mid-2002.

The map data is currently provided in the L-EST co-ordinate system for compatibility with other Estonian datasets (see next section), the fully functional server should offer the same data also in unprojected (latitude/longitude co-ordinates) and perhaps also in Mercator projection, traditionally used for nautical charts.

The system consists of a dedicated server computer with map database, web server, and F-Open server WMS server software.

In the map database are stored all electronic nautical charts that are currently valid for navigating in Estonian waters. Each chart is divided into several layers (navigational marks, soundings, navigational restrictions, etc.). Some of the layers (e.g. navigational marks) would be queryable, i.e. user can use the *GetFeatureInfo* interface of the WMS protocol for fetching information about the objects on those layers (see Figure 5.4).

The map layers would be collected into high-level divisions, based on the scale and purpose of the charts. The top layer displays the contours of the coastline of the Baltic Sea.

[1] http://www.vta.ee/

The structure of the layers can be seen in Figure 5.2 on page 49 (center part of the left panel).

The system contains also the WMS client software which is based on the *WMS Client Toolkit*, tailored to this particular project (direct connection to the Estonian Maritime Administration's WMS server, displaying of additional textual information about the charts, etc.). The service itself (the cartographic data) is indeed perfectly usable with any WMS client application.

The goals of this project are quite simple and could be achieved with other map servers (with proprietary technology). The use of the open WMS protocol opens up additional possibilities, for example the fictional information server of the yacht harbours of Estonia can use these nautical charts as a background for their own map layers, showing the information about the harbour facilities, hotels, or other information.

6.2 Data Exchange between Map Authorities

This section describes the pilot of the process of connecting the Geographical Information Systems of two big map producing state agencies, EMA and *Estonian Land Board (ELB)*[2]. EMA produces the nautical charts for Estonian waters, ELBis the producer of several large spatial datasets (Land Cadastre, Estonian Base Map, Estonian Basic Map and others). Although the products of the two organizations are principally different, they do share some common objects, such as coastline, landmarks usable for navigation, etc.

Such situation mandates the checking and validating each other's data, as the resulting maps need to be consistent and not misleading. ELB and EMA are currently changing data occasionally on compact discs. The aim of this project is to provide instant access to the spatial databases of the other authority, eliminating the need of physically transporting the data and providing on-line access to the most recent data.

The structure of the data, used by the agencies is quite different, let alone the tools that are used for producing the maps. These factors make the interoperability between these systems on the record (map feature) level very difficult and costly. Therefore the WMS protocol was chosen as an intermediating agent, the WMS clients run in ubiquitous web browsers, providing the raster image of the area of interest. When the cartographer has layered images from two map servers on top of each other (either by using a cascading server or by taking advantage of the composite image technology of the WMSClient or WMS Client Toolkit-based application), he can immediately spot any differences between the datasets. He can then make the *GetFeatureInfo* request for getting the GML representation of the feature object from the distant server. The GML file is then analyzed and compared with existing data. The process comes to the end with patching the map with new data that is converted to suitable form.

The pilot was conducted during the end of 2001 and the beginning of 2002 for testing the basic technology (setting up WMS servers, testing the connectivity, etc.).

ELB provided a test server, based on the MapServer. During the test period this server was populated with data about the Estonian administrative units and a piece from the Estonian Base Map[3]. Estonian Maritime Administration used the public server prototype (see previous section), based on the F-Open server software, containing some nautical charts. The data in both servers were in the L-EST co-ordinate system. An example of the composite map from these two map servers is shown in Figure 5.2 on page 49.

[2] http://www.maaamet.ee/

[3] For practical work the Estonian Basic Map will be used.

The pilot was carried out on the public Internet without any access restrictions, the production system would certainly need strict access restrictions for allowing only certified personnel to access the databases.

The results of the pilot were positive but the completion of the whole project have yet to happen. The reasons for delaying the project are more administrative than technical, mostly dealing with the ownership of the data, the organization of process of modifications in case differenced are spotted between a nautical chart and the Estonian Basic Map, and other similar questions.

There exist additional problems, mostly related to data quality and different co-ordinate systems being used, that need attention.

The L-EST system is used for the Estonian Basic Map and most other maps in Estonia. Nautical charts are printed in Mercator projection with varying standard parallels for different chart sheets, ENCs contain geographical co-ordinates. Co-ordinate system transformation is therefore needed for spatial data, imported from outside (i.e. other sources than the main database for given organization) which can cause some serious, hard-to-catch errors if not carried out carefully. The problem grows larger when data more than two different co-ordinate systems need to be integrated.

Every agency is responsible for the accuracy of the maps they produce. The requirements for accuracy for nautical charts are precisely defined and stricter than for other maps. EMA wants to be absolutely sure about the quality of the features they get from other sources, as they would be legally liable for ship accidents, caused by errors on published charts.

A proposal for building a more complex system around the FRED map editor has been made for the EMA by the *R-Süsteemid* company. This system would integrate into the process of preparing nautical charts in the S-57 format, providing a raster image (obtained from the WMS server) as a background to vector data in S-57 format. Using the WFS for accessing the features may also prove practical in the future.

6.3 The Registry of Assets of RMK

R-Süsteemid created a web-based registry of assets for Riigimetsa Majandamise Keskus /State Forest Management Agency/ (RMK) at the beginning of the year 2002. The application used a MySQL[4] database for storing the information and contained various forms for processing and querying the data using the web browser. The database contained all assets of the RMK—computers, vehicles, land parcels, leases, etc.

The registry includes several types of properties that contain spatial components: land parcels, buildings, etc., leading to the idea of graphical representation of such information as a map (originally the interaction with the registry was only via alphanumerical forms). The choice of the WMS standard was quite natural, as the registry was already built for WWW environment. The standard protocol would also provide beneficial in the future when background maps can be fetched from other servers (e.g. Estonian Basic Map).

At present time a prototype of such system has been produced for demonstrating the feasibility of this approach, containing the original registry program and database, a spatial database, F-Open server WMS server and a special WMS client application, created with the WMS Client Toolkit. The prototype includes data for only land parcels that are located in Rapla county.

The original registry database contained insufficient information for visualizing land parcels, they were identified by only one co-ordinate pair and the cadastral identifier pointing to the Land Cadastre, held by the ELB. The geometries of the land parcels were copied from

[4]http://www.mysql.com/

the Land Cadastre to a new PostGIS database[5], character (attribute) data originated from the registry of assets of RMK. PostGIS database was chosen because it is free software (good for minimizing the cost of a prototype system) and it supports the OGC *Simple Features for SQL* ([42]) data model (see also Section 4.3.2). F-Open server loads data from spatial database, attribute database (assets of RMK) and uses a specially created presentation library for visualizing the data.

Currently two special operations have been added to the asset registry: WMS client is opened with the map, showing the location of the land parcel when clicking on a button in parcel's (textual) information screen. The other innovation performs the opposite function—clicking on the map can bring up the appropriate properties of the land parcel from MySQL database. The latter function is based on the *GetFeatureInfo* request, the location of the mouse pointer is transferred into co-ordinates, map server finds the object(s) and returns the feature data, encoded in GML. The link into the registry program is done using the identifier inside the feature data, returned by the WMS server.

The public demonstration of the application is available at `http://r-systems.ee:7007/rmk/WMS/rapla.html` (screenshot is shown in Figure 5.3 on page 51). The map server contains four layers: RMK land parcels and also the full land cadastre information for Rapla county, forest compartments, and forest partitions for Vardi forest district (samples of real data provided by RMK). The other three layers serve as additional background information for giving context for the RMK land parcels. The map of administrative units from the test server of ELB (see previous section) is used as the bottommost layer (not shown in Figure 5.3) for a proof-of-concept for interoperability.

6.4 Conclusion

The projects described in this chapter indicate that the time is ready for using the Web Map Service (WMS) protocol for solving real world problems. The OpenGIS Consortium geospatial services standards can turn the web-based Geographical Information Systems into a realty—you can use your maps everywhere, no matter where you are.

The public Internet map servers are not new players on the GIS arena and they are certainly used not only for amusement purposes but also for real work. Using the WMS-based public map servers would benefit all users, as they receive new qualities, such as the exact georeferencing of the map images, extensive metadata, and most importantly the great interoperability of the WMS servers. These enhancements build up from the ground set up by earlier map servers, still preserving their strong points (cross-platform access via web browsers, simplicity).

When the on-line system for spatial data exchange between Estonian Maritime Administration and Estonian Land Board is completed, it will certainly increase the quality of maps, produced by those agencies. The system can be easily extended later to exchange data between other map-makers as well.

The combination of the asset registry of the RMK with the WMS has been successfully demonstrated. Besides the impacts to the registry project (increased efficiency in managing the land parcels) it has shown the feasibility of this approach for other similar projects. The task of adding spatial dimension to an existing database (turning an information system into a GIS) is usually not trivial but the results are worth the trouble. The WMS protocol can provide an easy and effective solution for web-based databases (also for others but more work is needed for proprietary database interfaces) for the 'spatialization' problem.

[5]Some sort of service will be used in the future for fetching cadastral data from the server of the Estonian Land Board.

CHAPTER 7

Conclusions

It is widely recognized that standards are the key to address interoperability: spatial data handling provides enough challenges and there is no need for creating artificial obstacles to the process of using geospatial data. In this work I have advocated the importance of using open standards for maximizing the interoperability in hierarchical and peer-to-peer systems. Using open standards that are approved by international organizations standards is strongly recommended for every GIS application.

It has been a pleasure to work in the front line of the GIS development. The current thesis is one of the first publications (together with the previous technical report which was co-authored by me ([29])) that covers the OpenGIS services in the context of present Estonian reality.

The results of my research are summarized in the list below (more detailed discussions of these results are given at the end of corresponding chapters):

- An overview of two basic methods of interoperability of Geographical Information Systems, the exchange of files and the use of network based data services was given in the third chapter. Introduction to the interoperability was provided together with the descriptions of several widely used formats of spatial data files. Comparisons of the described formats and suggestions for choosing a file format were also provided in this chapter.

  While there are many different formats and methods used for communication and data exchange in GIS community, quite a large number of these do not qualify as robust, adequate or do not have open specifications available. However, there exist high quality open standards for both services and file formats. Open standards that are created in co-operation of many participants should be preferred to proprietary, one-company solutions, because internationally approved standards have adequate support for metadata, their specifications are available for third-party implementations, and they have usually more comprehensive and universal data models. From the file formats covered in this thesis I suggest using the DIGEST standard, as it provides all previously mentioned qualities (S-57 has very similar capacity but it is more specialized). GML may prove adequate for applications that carry out only simple operations on data (GML does not support topology in the current version) and make extensive use of the XML technology.

  As Geographical Information Systems are migrating from the desktop to a networked environment the services will become more important forms of obtaining spatial data. Therefore the focus was also on services in the chapter of interoperability. Most attention was given to the family of web-based geospatial services provided by the Open

GIS Consortium. Services are relatively new method for providing interoperability in the GIS community but they will undoubtedly become more important in the future.

- Chapter 3 introduced two specifications of the OGC services—Web Map Service and Web Feature Server. Chapters 4 and 5 moved from formal specifications to practical realizations of the WMS standard, providing descriptions of several WMS server and client applications, respectively. A more in-depth view of one WMS server program, F-Open server, was also provided for better understanding of the WMS protocol and software.

- The increased interoperability of Geographical Information Systems that resulted from the adoption of an open standard (the OGC WMS—a service for providing georeferenced map images together with descriptive metadata in the WWW environment) was demonstrated in this thesis.

  Three examples of using the Web Map Service protocol for solving practical problems were provided in Chapter 6. These projects demonstrated clearly that open standards are very useful for building innovative software applications that can easily exchange data with other systems.

I would also like to point out some trends and problems for future development. The work on improving existing standards and standardization new areas should definitely continue. Standard organizations should do even more co-operation for ensuring the compatibility of different specifications.

Geospatial data services are getting more and more attention and are becoming important data sources these days. Hopefully there will be a network of WMS servers with affiliations to other services (geocoders, gazetteers, geoparsers, etc.) for provision of miscellaneous geographical data. WFS and other OGC specifications will gain importance in their respective segments, providing transparent access to omnipresent feature data stores. Some of these services would be publicly accessible but most will probably be private servers with restricted access for employees or paying customers (similar to the present situation with data files).

For wider acceptance of WMS (and other standards) it is very important that the standard is accepted early by major data providers. In Estonia it would mean that e.g. the Estonian Land Board should set up a WMS server for providing full coverage data for Estonia. The map server should be either public or semi-public (access restricted for specific organizations) and should contain a good large scale map (the Estonian Basic Map or Land Cadastre, for example). The WMS protocol would guarantee the usability of such services as a background information in many map servers that provide some map layers which are informative but may look unnecessarily sketchy without additional background. The on-line access to central WMS server would also mean that the latest version of database is used, the owner of some smaller server can concentrate to the quality of his data without worrying much about the timeliness of the background information.

Glossary

Abbreviated Terms

ANSI American National Standards Institute

CAD Computer Aided Design

CGI Common Gateway Interface

CSDGM Content Standard for Digital Geospatial Metadata

DBMS Database Management System

DCP Distributed Computing Platform

DIGEST Digital Geographic Information Exchange Standard

DGIWG Digital Geographic Information Working Group

ECDIS Electronic Chart Display and Information System

ELB Estonian Land Board

EMA Estonian Maritime Administration

ENC Electronic Navigational Chart

EPSG European Petroleum Survey Group

ESRI Environmental Systems Research Institute

FACC Feature Attribute Coding Catalogue

GIF Graphic Interchange Format

GIPSIE GIS Interoperability Project Stimulating the Industry in Europe

GIS Geographical Information System

GML Geography Markup Language

HTML Hypertext Markup Language

HTTP Hypertext Transfer Protocol

IEC International Electrotechnical Commission

IHO International Hydrographic Organization

IMO International Maritime Organization

ISFF Intergraph Standard File Format

ISO International Organization for Standardization

JPEG Joint Photographic Expert Group

ODBC Open Database Connectivity

OGC Open GIS Consortium

PHP PHP Hypertext Preprocessor

PNG Portable Network Graphics

RMK Riigimetsa Majandamise Keskus /State Forest Management Agency/

SDTS Spatial Data Transfer Standard

SGML Standard Generalized Markup Language

SLD Styled Layer Description

SRS Spatial Reference System

TIFF Tag Image File Format

TIGER Topologically Integrated Geographic Encoding And Referencing System

WBMP Wireless Bitmap

WCS Web Coverage Server

WFS Web Feature Server

WMS Web Map Service

WRS Web Registry Service

WWW World Wide Web

XML Extensible Markup Language

XSL Extensible Stylesheet Language

Glossary

Attribute A trait, quality or property describing a geographical feature.

Co-ordinate system A numerical method for representing locations on the Earth's surface.

Data dictionary The description description of the basic organization of a database. A data dictionary typically contains a list of all files in the database, the names and types of all records and metadata about the records and restrictions. Also known as *feature catalogue*.

Dataset An identifiable collection of related data. One dataset (also *data set*) contains specific data (and accompanying metadata) from a certain area.

Feature A set of points, lines or polygons in a spatial database that represents an abstraction of a real-world phenomenon (entity). The terms *feature* and *object* are often used synonymously.

Feature catalogue See *data dictionary*.

GIS Information system dealing with information concerning phenomena associated with location relative to the Earth.

Geomatics A technology and service sector focussing on the acquisition, storage, analysis, dissemination and management of geographically referenced information for improved decision-making.

Geocoder A system for assigning geographic co-ordinates to a street (postal) address.

Metadata Data describing and documenting data.

Object See *feature*.

Service A service is a capability which a service provider entity makes available to a service user entity at the interface between the entities.

Topology The properties of spatial configuration which remain invariant under continuous (in all directions) transformation. Also the branch of mathematics that investigates those propeties.

Bibliography

[1] Alexander, L. Electronic Charts: What, How, and Why?, June 2000.
URL http://www.caris.com/S-57/electronic_charts.html 19

[2] Association for Geographic Information. The GIS Dictionary, 1999.
URL http://www.geo.ed.ac.uk/agidict/welcome.html 4, 5

[3] Aunap, R. Eestis kasutatavad koordinaatsüsteemid. Coordinate Systems used in Estonia, April 2001.
URL http://www.geo.ut.ee/~raivo/ESTCOORD.HTML 8

[4] Booth, C. J. (Editor) *The New IEEE Standard Dictionary of Electrical and Electronics Terms*. IEEE Std 100-1992. The Institute of Electrical and Electronics Engineers, Inc., fifth edition, 1993. ISBN 1-55937-240-0. 3

[5] Borland Software Corporation. *dBASE .DBF File Structure*, 7 1998.
URL http://community.borland.com/article/0,1410,15838,00.html 16

[6] Cargill, C. F. Why Are We Doing This? *Computer*, *volume 34*, 10 pp. 116–117, October 2001. ISSN 0018-9162. 14

[7] Conforti, F. The new DGN file format is the foundation for revolutionary changes to MicroStation. *MicroStation Manager*, June 2001. ISSN 1057-9567.
URL http://www.msmonline.com/jun01/cover.htm 18

[8] CubeWerx Spatial Warehouse Architecture.
URL http://www.cubewerx.com/main_en/system_architecture.html 37, 46

[9] The Digital Geographic Information Exchange Standard (DIGEST). Part 1. General Description. Edition 2.1, Digital Geographic Information Working Group, September 2000.
URL http://www.digest.org/documents/Part1_GeneralDescription.zip 20

[10] Discussion Paper: OpenGIS Styled Layer Descriptor Draft Candidate Implementation Specification. Version 0.7.0. OpenGIS® Implementation Specification OGC 01-028, Open GIS Consortium Inc., February 2001.
URL http://www.opengis.org/techno/discussions/01-028.pdf 30

[11] Egenhofer, M. J. and Mark, D. M. Naive Geography. In A. U. Frank and W. Kuhn (Editors) *Spatial Information Theory*, volume 988 of *Lecture Notes in Computer Science*, pp. 1–15. Springer, Berlin (Germany), 1995.
URL http://www.geog.buffalo.edu/ncgia/i21/ng/NG51.html 4

[12] ESRI Shapefile Technical Description. An ESRI White Paper, Environmental Systems Research Institute, Inc., July 1998.
URL `http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf` 15

[13] Federal Geographic Data Committee. Content Standard for Digital Geospatial Metadata. FGDC-STD-001-1998, National Spatial Data Infrastructure, Washington (DC, USA), June 1998.
URL `http://www.fgdc.gov/standards/documents/standards/metadata/v2_0698.pdf` 9

[14] Filter Encoding Implementation Specification. Version 0.0.7. OpenGIS® Implementation Specification OGC 01-067, Open GIS Consortium Inc., October 2001.
URL `http://www.opengis.org/techno/RFC13a.pdf` 32, 33

[15] Frank, A. U. and Mark, D. M. Language Issues for GIS. In Maguire *et al.* [33], chapter 11, pp. 147–163.
URL `http://www.wiley.co.uk/wileychi/gis/Volume1/BB1v1_ch11.pdf` 4

[16] Fuller, S. and Cox, T. Anatomy of a Forward-Looking Open Standard. *Computer*, *volume 35*, 1 pp. 140–141, January 2002. ISSN 0018-9162. 14

[17] Galdos Systems Inc. Why GML?
URL `http://www.galdosinc.com/technology-whygml.html` 21

[18] Gatrell, A. C. Concepts of Space and Geographical Data. In Maguire *et al.* [33], chapter 9, pp. 119–134.
URL `http://www.wiley.co.uk/wileychi/gis/Volume1/BB1v1_ch9.pdf` 5

[19] Geographic Information—Terminology. ISO Committee Draft CD 19104, ISO/TC 211, November 1999. 13, 25

[20] Geography Markup Language (GML) 2.1.1. OpenGIS® Implementation Specification OGC 02-009, Open GIS Consortium Inc., April 2002.
URL `http://www.opengis.net/gml/02-009/GML2-11.html` 21, 22

[21] Halaris, G., Hadzilakos, T., Kavouras, M., Panopoulos, G., Paraschakis, I., Sellis, T., Tsoulos, L., and Zervakis, M. Interoperability and Definition of a National Standard. Issues of Interoperability, Database Integration and Definition of a National Standard for the Exchange of Digital Geographic Data. 1998. GIS PlaNET 98 International Conference on Geographic Information, Lisbon (Portugal).
URL `http://www.dbnet.ece.ntua.gr/~timos/geodb/epaged.pdf` 14, 21

[22] Hecht, L. Stand up for Standards. *GEOEurope*, *volume 9*, 9 pp. 38–42, September 2000. ISSN 0926-3403.
URL `http://www.geoplace.com/ge/2000/0900/0900stn.asp` 27

[23] International Hydrograhic Organization. *IHO ECDIS Presentation Library User's Manual*. International Hydrographic Bureau, Monaco, 3.2 edition, May 2000.
URL `http://www.iho.shom.fr/general/ecdis/download/pslb03_2.pdf` 19

[24] International Hydrographic Organization. *Catalogue of the IHO Pulications 2002–2003*. International Hydrographic Bureau, Monaco.
URL `http://iho.shom.fr/publicat/free/files/P402En.pdf` 18

[25] International Hydrographic Organization. *IHO Transfer Standard for Digital Hydrographic Data. Edition 3.0.* International Hydrographic Bureau, Monaco, November 1996. 18, 20

[26] Krechmer, K. The Need for Openness in Standards. *Computer*, *volume 34*, 6 pp. 100–101, June 2001. ISSN 0018-9162. 14

[27] Krusberg, P. *Ruumiandmete töötluse võimalusi (Eesti baaskaardi näitel).* Magistritöö, Tallinna Tehnikaülikool, Tallinn (Estonia), November 1997. In Estonian. 5

[28] Kuus, H. *Functions for Handling Digital Maps.* Diploma Project, Tallinn Technical University, Tallinn (Estonia), June 1998.
URL `http://home.anet.ee/~hkuus/diploma/index.html` 39

[29] Kuus, H., Krusberg, P., and Rebane, R. Geoinformaatika standardid. Ülevaade, Riigi Infosüsteemide Osakond, March 2002. In Estonian.
URL `http://ats.riik.ee/amphora/home/projektid/gis/Standardid/HTML/index.html` 27, 60

[30] Laurini, R. and Thompson, D. *Fundamentals of Spatial Information Systems.* Number 37 in The A.P.I.C. Series. Academic Press, London (Great Britain), San Diego (USA), New York (USA), 1996. ISBN 0-12-438380-7. 3, 5, 8

[31] Limp, F. Don't Hit Warp Speed with the Wrong Equipment. *GEOEurope*, *volume 8*, 12 pp. 1822–42, December 1999. ISSN 0926-3403.
URL `http://www.geoplace.com/ge/1999/1299/1299wm.asp` 26

[32] Maguire, D. J. An Overview and Definition of GIS. In Maguire *et al.* [33], chapter 1, pp. 9–20.
URL `http://www.wiley.co.uk/wileychi/gis/Volume1/BB1v1_ch1.pdf` 3

[33] Maguire, D. J., Goodchild, M. F., and Rhind, D. W. (Editors) *Geographical Information Systems: Principles and Applications.* Longman, London (Great Britain), 1991. ISBN 0-582-05661-6. 66, 67

[34] Maling, D. H. Coordinate Systems and Map Projections for GIS. In Maguire *et al.* [33], chapter 10, pp. 135–146.
URL `http://www.wiley.co.uk/wileychi/gis/Volume1/BB1v1_ch10.pdf` 6

[35] MapServer Homepage.
URL `http://mapserver.gis.umn.edu/` 35

[36] Mark, D. M., Chrisman, N., Frank, A. U., McHaffie, P. H., and Pickles, J. The GIS History Project. In *Proceedings of the UCGIS Summer Assembly.* Bar Harbor (Maine, USA), June 1997.
URL `http://www.geog.buffalo.edu/ncgia/gishist/bar_harbor.html` 1

[37] Meitner, M. J., Cavens, D., and Sheppard, S. R. Academic Metadata Standards: Getting Compliance Without Enforcement. In *ESRI User Conference Proceedings.* July 2001.
URL `http://www.esri.com/library/userconf/proc01/professional/papers/pap934/p934.htm` 9

[38] *MicroStation 95 Reference Guide*, chapter 18.
URL `http://gdal.velocet.ca/projects/dgn/ref18.pdf` 17, 25

[39] Moore, T. Coordinate Systems, Frames and Datums, 2000.
URL `http://granby.nott.ac.uk/~iszwww/coord1.htm` 7

[40] Newell, R. G. and Sancha, T. L. The Difference Between CAD and GIS. Smallworld Technical Paper No. 2, GE Smallworld.
URL `http://emea.smallworld.co.uk/support/techpaper/tp2.html` 15

[41] Open GIS Consortium Inc. Overview of OpenGIS Implementation Specifications, August 2001.
URL `http://www.opengis.org/info/gisworld/20010911.TS.SpecOver.htm` 27

[42] OpenGIS® Simple Features Specification For SQL. Revision 1.1. OpenGIS Project Document OGC 99-049, Open GIS Consortium Inc., May 1999.
URL `http://www.opengis.org/techno/specs/99-049.pdf` 41, 59

[43] Painho, M., Peixoto, M., Cabral, P., and Sena, R. WebGIS as a teaching tool. In *ESRI User Conference Proceedings*. July 2001.
URL `http://www.esri.com/library/userconf/proc01/professional/papers/pap910/p910.htm` 26

[44] Prothman, B. Meta data. *IEEE Potentials*, *volume 19*, 1 pp. 20–23, February/March 2000. ISSN 0278-6648. 8

[45] Robinson, A. H., Morrison, J. L., Muehrcke, P. C., Kimerling, A. J., and Guptill, S. C. *Elements of Cartography*. John Wiley & Sons, New York (U.S.A), Chichester (Great Britain), Brisbane (Australia), 1995. ISBN 0-471-55579-7. 5, 6

[46] Roy, A. Web Mapping Solutions.
URL `http://www.gisdevelopment.net/technology/gis/techgi0037pf.htm` 26

[47] Rüütel, M. *Meremõõdistussüsteemi tarkvara loomine*. Diplomitöö, Tallinna Tehnikaülikool, Tallinn (Estonia), June 1997. In Estonian. 39

[48] Scheuermann, W. What is ECDIS and what can it do?, October 1996.
URL `http://www.sevencs.com/ecdis/whatisecdis.htm` 19

[49] Snellen, I. ICTs, Bureaucracies, and the Future of Democracy. *Communications of the ACM*, *volume 44*, 1 pp. 45–48, January 2001. ISSN 0001-0782. 1

[50] Snyder, J. P. *Map Projections—A Working Manual*. Number 1395 in U. S. Geological Survey Professional Paper. United States Government Printing Office, Washington (DC, USA), 1987. ISBN 9998 605067. 8

[51] STASYS Limited. WGS 84—World Geodetic System 1984.
URL `http://www.wgs84.com/` 6

[52] The OpenGIS™ Abstract Specification Topic 0: Abstract Specification Overview. Version 4. OpenGIS™ Project Document Number 99-100r1, Open GIS Consortium Inc., June 1999.
URL `http://www.opengis.org/techno/abstract/99-100r1.pdf` 22

[53] The OpenGIS™ Abstract Specification Topic 5: Features. Version 4. OpenGIS™ Project Document Number 99-105r2, Open GIS Consortium Inc., June 1999.
URL `http://www.opengis.org/techno/abstract/99-105r2.pdf` 32

[54] Theobald, D. M. Understanding Topology and Shapefiles. *ArcUser*, *volume 4*, 2, April-June 2001.
URL `http://www.esri.com/news/arcuser/0401/topo.html` 5, 15, 16

[55] United States Census Bureau. TIGER® Overview, 2001.
URL `http://census.gov/geo/www/tiger/overview.html` 14

[56] Warmerdam, F. DGNLib: a Microstation DGN (ISFF) Reader.
URL `http://gdal.velocet.ca/projects/dgn/index.html` 25

[57] Warmerdam, F. Microstation DGN Feature Representation.
URL `http://gdal.velocet.ca/projects/dgn/representation.html` 18

[58] Web Feature Server Specification. Version 0.0.14. OpenGIS® Implementation Specification OGC 01-065, Open GIS Consortium Inc., October 2001.
URL `http://www.opengis.org/techno/RFC13.pdf` 31

[59] Web Map Service Implementation Specification. Version 1.1.1. OpenGIS® Implementation Specification OGC 01-068r3, Open GIS Consortium Inc., January 2002.
URL `http://www.opengis.org/techno/specs/01-068r3.pdf` 14, 28, 30, 43

[60] Winter, S. and Frank, A. U. Topology in Raster and Vector Representation. *GeoInformatica*, *volume 4*, 1 pp. 35–65, March 2000. ISSN 1384-6175.
URL `ftp://ftp.geoinfo.tuwien.ac.at/winter/winter00topology.pdf` 5

[61] Winter, S., van der Vlugt, M., and Frank, A. U. Open GIS and Interoperability in Europe. In *Proceedings of GIS Ostrava '99*. Areal VSB-TU Ostrava, Ostrava (Czech Republic), 1999.
URL `ftp://ftp.geoinfo.tuwien.ac.at/winter/winter99open.pdf` 12

[62] XML Schema. Part 1: Structures. W3C Recommendation, World Wide Web Consortium, May 2001.
URL `http://www.w3c.org/TR/xmlschema-1/` 22, 33