USING VARIO-SCALE DATA IN A WEB SERVICES SETTING

GRADUATION PLAN

ADRIE ROVERS

USING VARIO-SCALE DATA IN A WEB SERVICES SETTING

GRADUATION PLAN

MASTER OF SCIENCE IN GEOMATICS FOR THE BUILT ENVIRONMENT

NAME:	ADRIE ROVERS
STUDENT NUMBER:	1503391
TELEPHONE NUMBER:	+31610965508
EMAIL ADDRESS:	A.ROVERS@STUDENT.TUDELFT.NL
FIRST MENTOR:	DR. IR. MARTIJN MEIJERS
SECOND MENTOR:	PROF. DR. IR. PETER VAN OOSTEROM

JUNE 2016

CONTENTS

1 INTRODUCTION 1
- 1.1 Context . . . 1
- 1.2 Problem statement . . . 3
- 1.3 Scientific relevance . . . 3

2 RESEARCH BACKGROUND 4
- 2.1 Theoretical framework . . . 4
- 2.2 Related work . . . 6

3 OBJECTIVES AND RESEARCH QUESTIONS 9
- 3.1 Objectives . . . 9
- 3.2 Research questions . . . 9
- 3.3 Research scope . . . 10

4 METHODOLOGY 11
- 4.1 Design science research . . . 11
- 4.2 Planning . . . 11
- 4.3 Tools . . . 12
- 4.4 Data . . . 13
- 4.5 Meetings . . . 13

A INITIAL CONCEPTUAL DESIGN 17

1 INTRODUCTION

1.1 CONTEXT

Geographic information is used to solve a diversity of problems in various application areas. Areas of use are decision making (Sugumaran and Degroote, 2010), water management (Lyon, 2002), urban planning (Geertman et al., 2013), route finding and many more (Lemmens, 2011). Depending on the application, different spatial models are used to represent reality. A model uses abstraction to represent the fundamental concepts that are relevant to the application domain. We can not capture every aspect of our world because it is to complex, and the model allows us to focus on the relevant details instead.

In these models multiple dimensions can be represented. They can include 1, 2 or 3 spatial dimensions. For ages cartographers used 2D maps to model the shape of the earth, and more recently 3D models are being used for the analysis, simulation and visualisation of our environment.

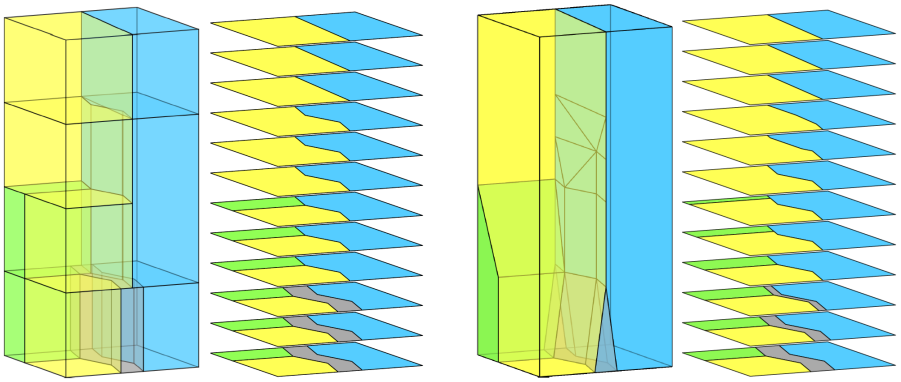
An important aspect concerning geographic information is the amount of detail that is captured in the model. Is a road represented as a line or as a polygon? Do we simplify certain features or are they not relevant for the application domain and not modelled at all? These considerations are commonly captured in discrete levels of detail, like in the CityGML standard where five levels are defined (Gröger and Plümer, 2012).

This level of detail is directly related to the concept of scale in 2D maps. The scale of a map is the ratio between the distances on the map and those in reality. Because of its fixed size, the information that can be conveyed on a map is limited. The amount of detail decreases as the scale gets smaller. Think of a map of the world, where individual countries can be distinguished, as opposed to a map of the city, where road networks and neighbourhoods can be perceived.

Different map scales are commonly maintained for different purposes. For each fixed scale a separate layer of geographic information is stored (Meijers, 2011b). However, redundancy occurs as some features might exist at multiple scale levels. In addition, consistency is difficult to maintain because changes on one scale level should propagate to the next.

1.1.1 Vario-scale data

This brings us to the concept of vario-scale data. Instead of storing separate layers for each discrete scale level, a spatial model could also describe a continuous level of detail. Such a model is described in van Oosterom and Meijers (2013); van Oosterom et al. (2014), where scale is represented as a 3rd dimension. A generalisation process is used on a 2D base map and the results are stored in a single 3D structure, the so called Space Scale Cube or SSC (Figure 1.1). This model guarantees consistent data storage and



(a) The classic SSC. **(b)** The smooth SSC.

Figure 1.1: The Space Scale Cube: A single 3D model is maintained instead of storing separate maps for each scale level. Adapted from van Oosterom et al. (2014).

prevents redundancy. It is important to note that because level of detail is variable both space and scale become selective when querying the data.

The suitability of an object for a certain level of detail is determined by an importance value that is assigned to the objects. This model is depicted in Figure 1.1a. The implementation of such a model is realised by storing 2-dimensional edges and faces with their importance value and topological relationships in the database.

An alternative approach is depicted in Figure 1.1b. The discrete objects gradually fade or aggregate in the 3rd dimension, offering smooth level of detail, something which can not be achieved with discrete maps. This continuous change requires that objects are represented as polyhedrons in storage. Visualisation of a map is achieved by intersecting the SSC. The GPU can be used for this purpose, as implemented in Driel (2015).

A related model, where level of detail is integrated as a 4th dimension, is the 4D point cloud (van Oosterom et al., 2015). A point cloud is a collection of x,y,z coordinates representing the surfaces and objects that exist in reality. The points are usually acquired by Lidar laser scanners but this can also be done by means of photogrammetry. In this model, level of detail controls the density of the point cloud. This could be used to retrieve an initial overview of the scene when fast feedback is needed and to show more detail when the user zooms in on an object.

Vario-scale maps and 4D point clouds can bring new opportunities for the way in which spatial data is used. However, such models also present new challenges in terms of storage, access and dissemination.

1.1.2 Using vario-scale data in a web services setting

The usage of geo-information is shifting from centralised to distributed system environments. Worboys and Duckham (2004, p.266) define a distributed system as an information system where multiple components, connected through a communication network, cooperate in achieving a common task. A distributed environment regarding geo-data is commonly referred to as the Geo-Information Infrastructure or GII. Van Oosterom et al. (2000, p.10) point out several advantages: First of all, data can be maintained at the source. The need for users to maintain their own versions of the dataset

disappears and the data is always kept up-to-date. Secondly, subsets of the data can be retrieved, allowing fair pricing models. And thirdly, data becomes more accessible.

Current distributed systems are commonly based on the client-server architecture. This architecture is characterised by a clear separation of responsibilities (Worboys and Duckham, 2004, p.267). A server is a component that provides particular resources or processing capabilities, termed as services, and a client in turn is a component that makes use of these services. When the Internet is used as the communication network we can thus speak of 'web services'.

1.2 PROBLEM STATEMENT

This brings us to the general problem when using data in a web services setting: A client can not hold all data in memory. Thus, relevant data has to be transferred from server to client. However, transferring data takes time and sometimes also costs can be involved for every byte that is send over the network. It is apparent that redundant data transfers should be avoided as much as possible.

In the scenario where separate geographic datasets are maintained for each discrete level of detail, redundant data transfers are unavoidable. Even though some objects might exist at multiple scale levels, level of detail is not selective. Requesting more detail leads to the retrieval of a complete new dataset for a selected geographic region.

That being said, having vario-scale data, the opportunity arises to reuse data that is already present on the client and retrieve only missing data from the server. However, it is not yet apparent how this can be achieved. Accordingly, this research aims in achieving efficient communication, without too many redundant data transfers, for vario-scale data in a web services setting.

1.3 SCIENTIFIC RELEVANCE

An efficient communication method would contribute to the continuing research on vario scale data by van Oosterom and Meijers (2013) and van Oosterom et al. (2014). They currently implemented three different communication methods for varioscale data in a web services setting, which are described in Huang et al. (2016). However, they state that the ability to reuse the data is currently limited. Savings mainly come from the last response, while additional reuse could be beneficial.

Also van Oosterom et al. (2015) could benefit, since the density and thus size of point clouds are ever increasing by the improvements in technology. This makes point clouds hard to handle. Sending a coarse overview first and only sending additional points when specifically needed, reusing the already send points, is relevant for limiting data traffic.

Furthermore, this question possibly applies to 4D spatio-temporal and 5D space-time-scale models as well (see van Oosterom and Stoter, 2010). This assumption is based on the fact that objects exist throughout time, just like throughout scale, and can possibly be reused in a similar manner.

2 RESEARCH BACKGROUND

This chapter gives a background for the research. Section 2.1 shortly describes the theoretical concepts that are appropriate to understand the subject. Section 2.2 discusses related work, which is important because it shows what already has been done and where this research can contribute.

2.1 THEORETICAL FRAMEWORK

As explained in the introduction, models are used to represent the world around us. To implement the model, data should be structured in such a way that it can be physically embedded in memory, ultimately stored as bits. In fact, multiple modeling stages take place during system development, as shown in Figure 2.1.

The application domain is reality, referring to aspects as they actually exist. The application domain model represents those aspects relevant to the users of the system, including requirements which the system should meet. The conceptual computational model formalizes these requirements. This is a high level model that facilitates communication between analysts, designers and users, independent of any implementation details (Worboys and Duckham, 2004, p.55). The organization of the data comes next, followed by the physical implementation of the model in memory.

Two main approaches can be distinguished for the conceptual modeling of geographic information. These are the field based and the object based approach (Molenaar and van Oosterom, 2009). In a field based model, space is fully partitioned and attributes are related to each position in the field. Temperature or land use are typical examples that can be efficiently modeled using this approach. An object based model does not partition the embedding space, it defines objects that can have thematic and geometric attributes instead.

A logical model describes the organization of the data using a specific data structure. For spatial data, a raster or vector representation is most commonly used. Molenaar and van Oosterom (2009) point out that raster structures are often associated with field models, just like vector representations with object models, but that these relations are not exclusive. Also, topological considerations take place in this stage. An example of a logical model could be the set of edges, faces and nodes with their topological relationships as captured in different tables.

This leads to the underlying implementation and physical representation of any model in computer systems. Storage structures, indexes and compression techniques are needed. The fundamental issue for storage is that computer memory is only 1-dimensional. This means that with the storage of spatial data some kind of mapping is needed. Gaede and Günther (1998) explain that there doesn't exist a mapping from n-dimensional to 1-dimensional space such that all objects that are close in reality are also

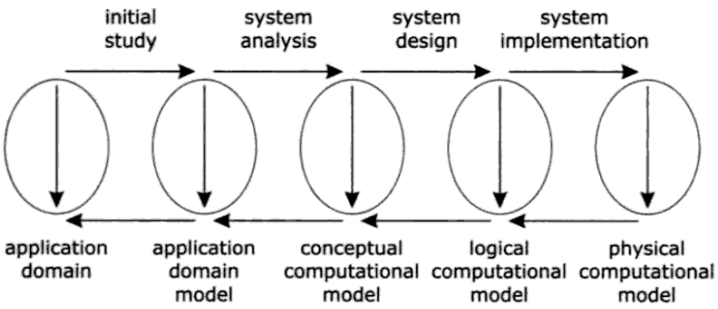


Figure 2.1: The modeling stages in system development. Taken from Worboys and Duckham (2004, p.137).

stored close in 1-dimensional space. An ordering can only be imposed on a single dimension. Therefore, n-dimensional, and thus spatial data, require different storage and access structures in order to be used efficiently.

2.1.1 Spatial access methods

Methods that support efficient storage and retrieval of spatial data are commonly referred to as spatial access methods. A spatial access method approaches both spatial indexing as clustering techniques (van Oosterom, 1999). An index helps in efficiently finding the right locations of data without having to perform a linear search. The index is a supplementary structure and therefore also requires space in memory. Clustering has the goal to group data that is likely to be requested together on the same or nearby disk pages in order to minimize access time. Clustering can be based on the organization of the index, but also space filling curves can be used for this purpose. A space filling curve basically maps the n-dimensional space to a single dimension while keeping some proximity between objects. This way common 1-dimensional indexing structures, such as the B-tree, can be used on the data (van Oosterom, 1999, p.388).

Spatial access methods can be divided into two main groups (Gaede and Günther, 1998; van Oosterom, 1999). The first group are methods that were developed for data that fits in main memory, allowing random access, and whereby the impact of clustering is not significant. These methods are thus not suited for large data sets that are stored on disk. This is addressed by a second group of methods, which take secondary memory usage into account. Because data is accessed on disk, clustering techniques become highly relevant.

Another way to distinguish spatial access methods can be found in the organization of their structure (Rigaux et al., 2001). A first group of methods builds an index by splitting up the space in which the objects are embedded. Each partition points to a location on disk and objects are mapped to these locations. Rigaux et al. call these space-driven methods. Optionally, this partitioning can be adapted to the distribution of objects. An example is to recursively divide the space depending on data density, like in the quadtree approach. As a consequence, objects are sometimes split up and distributed over disk pages. The great advantage of such methods is that the layout of the index is predictable. In an attempt to efficiently adapt to the irregular distribution of spatial data, a second group of structures are organized by grouping nearby objects together instead. The splitting of objects is avoided.

However, this comes at the expense of the simplicity of the index. These methods are referred to as data-driven.

The efficiency of a spatial access method may be expressed in terms of time complexity and space complexity (Rigaux et al., 2001; Worboys and Duckham, 2004). Time complexity refers to the time it takes to make a spatial selection. Searching should be optimized for the most important queries of the use case, such as point, range or nearest neighbour searches. Space complexity describes if the size of the index is reasonable with respect to the size of the dataset. A further requirement might be that the structure is dynamic, such that it efficiently adapts when deleting and inserting new objects.

A variety of spatial access methods already exist. Their development was largely driven by the characteristics of main and secondary memory (Gaede and Günther, 1998). However, requirements change when the access and querying of data shifts to a web services setting. Support for efficient communication is needed and therefore a method should also minimize retrieval of redundant data.

2.2 RELATED WORK

2.2.1 Classic SSC

The classic SSC is implemented in a web services setting as described in Huang et al. (2016). A topological data structure is maintained on the server. This is the compact tGAP (Meijers et al., 2009). It consists of nodes, edges and faces. Each primitive has its own importance range and therefore some edges may be associated with multiple faces during their lifespan. The decision was made to only explicitly store a reference to initial faces. As a consequence, edges sometimes point to an invalid face, which then has to be found via a separate table, the tGAP face-tree. An alternative is to keep multiple versions of an edge, but this would lead to redundancy in terms of storage since only the face references are different. Basically a trade off has to be made between redundancy and additional processing when deriving a map. However, savings in storage space turned out to be significant, while still having high performance. In a server-client architecture this also means that less data has to be transmitted.

For the communication between client and server Huang et al. developed three alternatives. All options retrieve data as topological primitives. Only edges and faces are send since nodes are already part of the edge geometry. The first option is stateless, i.e. each request is made independent of any previous responses. Reusing data that is already present on the client is thus not possible. The tGAP face-tree is consulted on the server and as a result each edge gets the right face pointers assigned before being send. The forming of geographic areas and rendering of the map is done on the client. To make a request the client only sends its viewport and gives the amount of objects that are preferred. A translation to importance values and thus level of detail is done on the server.

The second option aims in reusing data that is already on the client. Just like in the stateless approach edges and faces are retrieved. However, because edges are reused, also relevant parts of the face-tree need to be send in order to find the right faces for these edges. The client thus needs to do some additional processing. For the server to understand which data is

already present on the client, the client now sends a list of viewports. These include the new viewport, but also the viewports of previous responses. This information is used on the server to determine the delta of data that has to be send over the network. However, the more viewports, the more complicating the queries and thus more processing time is needed on the server. Time complexity increases and is linear with the number of previous responses (Huang et al., 2016, p.32). This reduces scalability. Huang et al. conclude that the savings that come from reusing previous responses are evened out by the face-tree that needs to be send in addition. No noteworthy bandwidth savings are achieved while extra processing time is needed.

The third option takes a completely different approach. Edges and faces are send in a sorted order, making it possible for the client to show a coarse overview and to incrementally update the map with more detail while the response is still being send. This is called progressive enhancement. Just like with the second option, caching is possible when the user is panning.

Huang et al. (2016) give directions that might be explored to make communication more efficient. A first possibility is to separate the geometry of the edges from the face pointers. Only geometry is cached. Instead of sending the face-tree it is then possible to only send new face pointers. Another option is to send the complete face-tree with the initial request. That way it is avoided that the same part of the face-tree is send over and over again during interactive use. Another idea is to make groups of data that are likely to be used simultaneously and let the client decide what to retrieve by making use of an index.

Meijers (2011a) proposes the use of a partition Fieldtree as additional data structure to make progressive data streaming more cache-friendly. A Fieldtree is a hierarchical structure with multiple levels. Each level in the tree consists out of multiple fields that form a planar partition of the domain. The fields of each level vary in size and are displaced with respect to the fields in other levels. Instead of using the entire domain, generalization takes place per field. Meijers suggests to use the fields as 3D blocks that partition the entire SSC and that can be retrieved and cached by the client. Because the Fieldtree partitions the space in a regular manner, it should be possible to compactly code the index (Meijers, 2011a). However, a custom generalization process is needed to create the fields.

2.2.2 Smooth SSC

The smooth SSC (Figure 1.1b) is a continuously generalised structure as an alternative model to the classic SSC (van Oosterom and Meijers, 2013). It is necessary to make an intersection on the SSC to derive a 2 dimensional map. Real time intersection can be achieved by utilizing the GPU, as implemented in Driel (2015). However, it is not possible to place the entire SSC in the memory of the GPU. A method is needed to subdivide the structure. The same goes when this data has to be transferred in a server-client architecture. Driel therefore pre-processes the SSC using an octree data structure. Overlapping nodes can be retrieved after a spatial selection. However, redundancy occurs because polyhedra are split up, introducing new vertices.

2.2.3 4D Point Clouds

The size of point clouds is ever increasing by the improvements in technology. This makes point clouds hard to handle. However, all this detail is not always needed. Think for example of a small mobile client, where screen resolution is limited. In addition, shipping of data from server to client can be costly, which makes the concept of level of detail highly relevant in a web services setting.

Van Oosterom et al. (2015) propose a data pyramid, consisting of multiple storage levels. Each level is subdivided in a different amount of data blocks, whereby each block approximately holds a constant number of points. Depending on the view position more or less blocks can be displayed and thus it is possible to vary data density for an increased performance.

In a following work, Martinez-Rubi et al. (2015) implemented a similar structure. A multi-resolution octree is created, which consists of a hierarchy of nodes. Based on this structure they developed a web viewer that visualizes AHN2 point cloud data.

However, the drawback of these approaches is that discrete transitions can be noticed, so called 'density shocks' (van Oosterom et al., 2015). This is a result of the varying point density between nearby blocks at different levels. Van Oosterom et al. propose to add an importance value to each point like in the vario-scale approach. This additional dimension can then be used to gradually increase point density based on the viewers position without introducing shocks. For efficient communication a method is needed to group points together, based on level of detail and geographic extent, that can be retrieved and cached by the client when needed.

3 OBJECTIVES AND RESEARCH QUESTIONS

Section 1.2 defined the overall goal of the research. The goal is to reach efficient communication, without too many redundant data transfers, for vario-scale data in web services setting. For this reason a method is needed to use the client cache and support retrieval of partial vario-scale data from the server. However, Section 2.2 made clear that the current methods are not yet optimal in terms of scalability and redundancy.

3.1 OBJECTIVES

In achieving our goal, this research tests if a generic data-driven spatial access method can be used for efficient retrieval of partial varioscale data over the web. The assumption is that efficient communication can be achieved by grouping objects with similar level of detail together. The following concrete objectives are defined:

1. *Group data that are likely to be used together into packages on the server; based on scale and geographic extent,*
2. *use client cache to reuse data,*
3. *and let the client retrieve additional packages using a generic spatial access method.*

3.2 RESEARCH QUESTIONS

The research questions follow from the objectives. The main research question of the thesis is as follows:

To what extent can a generic data-driven spatial access method support efficient retrieval of partial vario-scale data in a web services setting, and reuse of this data by means of caching?

In order to reach the objectives the following sub-questions have to be answered:

1. *What is the structure of the packages? More specifically, how should the multi-dimensional data be organised in the 1-dimensional memory of the database?*
2. *Which spatial access methods are suitable to be used as a starting point for the research?*
3. *How can a client keep track of the missing data? Or, how does a client know which packages it should request?*

4. *What should the cache strategy look like? In other words, how do we determine which data is most likely to be reused and which data should be discarded if the cache limit is exceeded?*

5. *How can we realise a scalable solution?*

6. *How are the performance measures? Is communication efficient compared to the current way of working?*

3.3 RESEARCH SCOPE

The focus in this thesis is mainly on reusing data packages on the client, and how the client cache can be used for this purpose. It investigates if based on current spatial access method principles, a client can implement the needed business logic to determine which partial data is needed. Not so much is this thesis about the other opportunities that varioscale data offer, like progressive enhancement or smooth content zoom (Huang et al., 2016). This means that the focus lays on constructing two successive maps, not on how this transition can be realised in a smooth manner.

Furthermore, the focus is on reactive data retrieval. This means that data is only requested from the server when it is directly needed and not yet in cache memory. In contrast, it should also be possible to design a proactive communication method, i.e. placing packages beforehand in the cache by predicting their relevance, possibly linked to what the user has done before. This is initially out of scope.

4 METHODOLOGY

4.1 DESIGN SCIENCE RESEARCH

The general methodology that will be followed in this thesis is based on the concept of 'design science research', as defined in Hevner and Chatterjee (2010, p.5). Instead of trying to develop understanding and theory by observing the world, the general aim is to gain knowledge through design. It's a method of research whereby new solutions ought to be found for existing problems by creating concrete artifacts.

The research will be conducted in an iterative manner, consisting out of two main activities;

- conceptual theory development assisted by a literature study,
- and the development of a prototype which is used to validate this conceptual theory.

The prototype will be developed for a specific use case, so that tests can be made with real data. During development more insights are gained about the problem area, which might lead to revising the conceptual theory. In the case that a problem is encountered, additional literature study will be done to find possible solutions. To be able to assess the prototype an evaluation and validation should be made based on objective measures. New insights can lead to readjustment of the conceptual solution once again. A development cycle takes place. Finally, an answer on the research question is given based on the functioning of the prototype, which can lead to recommendations or future work.

4.2 PLANNING

The activities defined in the methodology are part of a concrete planning (Gantt chart), which is given in Figure 4.1. It includes specific topics that should be part of the literature study. A first study on these aspects has been done and is part of this proposal. Furthermore, steps that need to be taken for the prototype development are included. Main activities are depicted in black and side activities in grey.

As discussed, theory development and prototype building take place in an iterative manner and therefore the chart is mainly used to give an indication when certain activities should be finished. Nonetheless, the planning shows two cycles that should at least take place. The first cycle should conclude with an initial benchmark before the P3 assessment. The prototype should have been objectively measured with a small dataset at this point. It is thereby not necessarily required to have finished the integration with the graphical client, retrieval could also be simulated. Feedback can then be received during the P3 and the conclusions can give direction to a second cycle.

The aim is to graduate in November. If it turns out that more time is needed after the P3, then the P4 and P5 can be shifted to December and January respectively. This would be a normal graduation period. The P3 has no fixed date. It is preferred that this is held at the end of July or the beginning of August. This way there is enough time to use any feedback for the P4. However, since the planning continues outside the official academic year, it should be discussed with the mentors if this is feasible.

The following dates are important:

3 June:	Submit graduation plan to the mentors and delegate of board of examiners.
10 June:	P2 presentation of 15 minutes.
Before P4 application:	P3 presentation of 15 minutes.
6 September :	Final application dates for P4.
1 week before P4:	Submit draft thesis to all mentors.
Week 39-40:	P4 presentation of 30 minute.
7 October :	Final application dates for P5.
1 week before P5:	Submit a hard copy of the final thesis to the main mentor, second mentor, third mentor and delegate of the board of examiners.
Week 44-45:	P5 presentation of 30 minutes.
1 week after P5:	Upload the final thesis (PDF) and final presentation slides (PDF) to the TU Delft repository.

4.3 TOOLS

Different tools and technologies will be used in this research. On the server side the PostgreSQL DBMS will be used together with the PostGIS extension. This extension makes it possible to store geographical data types. Python will be used as a general programming language. Initially, ideas will be tested in the Python environment by itself. To connect to the database Psycopg2 can be used. As an application framework Flask or Django are available. PHP can be used on the server as an alternative to Python. A connection with the database can then be made with the PDO abstraction layer. A local client can be set up using XAMPP or a related stack.

On the client we will use common web standards. These include HTML5, CSS3 and SVG. JavaScript is used for incorporating the business logic. User interaction and visualization can be achieved using the D3 JavaScript framework. WebGL can be used to access the GPU. As a message format (Geo)JSON will be used, since it is less verbose than XML/GML.

The Unified modelling language (UML) can be used for the development of the conceptual solution. The sequence diagram seems particularly useful. A first design is sketched, see Appendix A. The starting point is to separate requests for the index from requests for packages. It is assumed that this will simplify the implementation and make the solution more scalable.

4.4 DATA

The data comes from a specific use case. Depending on the progress, one or more use cases will be tested. Implementing more use cases will help in generalizing the theory. However, this might not be feasible in the time frame of the thesis. The classic SSC will be the first use case. A small dataset (ATKIS) is already received from the main mentor.

4.5 MEETINGS

Meetings should take place approximately once every two weeks with the main mentor and once every month with the second mentor. Alternatively, email communication can replace the meeting.

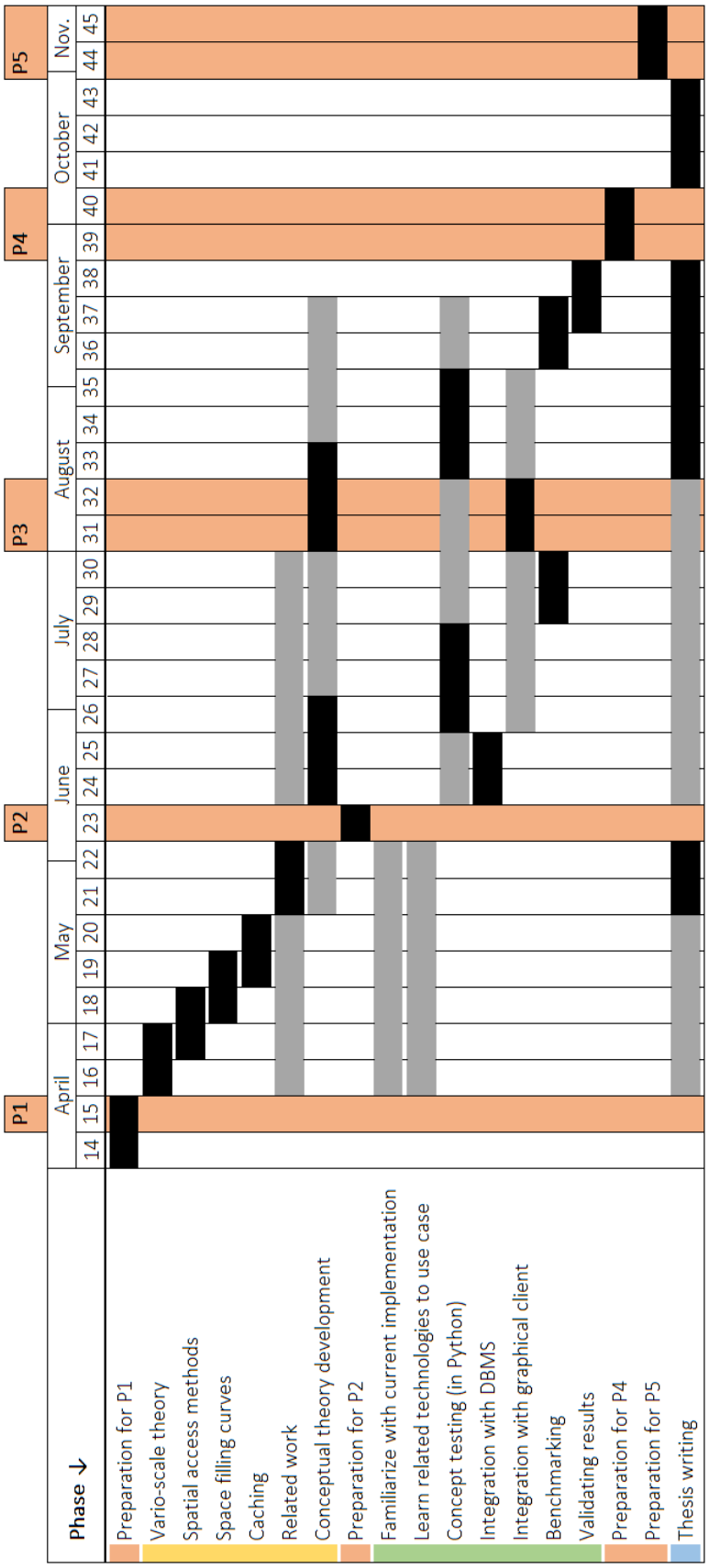


Figure 4.1: Gantt planning

BIBLIOGRAPHY

Driel, M. (2015). Real time intersections on space scale cube data.

Gaede, V. and Günther, O. (1998). Multidimensional access methods. *CSUR*, 30(2):170–231.

Geertman, S., Toppen, F., and Stillwell, J., editors (2013). *Planning Support Systems for Sustainable Urban Development (Lecture Notes in Geoinformation and Cartography)*. Springer, 2013 edition.

Gröger, G. and Plümer, L. (2012). CityGML – interoperable semantic 3D city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 71:12–33.

Hevner, A. and Chatterjee, S. (2010). *Design Research in Information Systems*. Springer US.

Huang, L., Meijers, M., Šuba, R., and van Oosterom, P. (2016). Engineering web maps with gradual content zoom based on streaming vector data. *ISPRS Journal of Photogrammetry and Remote Sensing*.

Lemmens, M. (2011). *Geo-information: Technologies, Applications and the Environment (Geotechnologies and the Environment)*. Springer, 2011 edition.

Lyon, J., editor (2002). *GIS for Water Resource and Watershed Management*. CRC Press, 0 edition.

Martinez-Rubi, O., Verhoeven, S., van Meersbergen, M., Schûtz, M., van Oosterom, P., Gonçalves, R., and Tijssen, T. (2015). Taming the beast: Free and open-source massive point cloud web visualization. In *Capturing Reality Forum 2015, 23-25 November 2015, Salzburg, Austria*. The Servey Association.

Meijers, M. (2011a). Cache-friendly progressive data streaming with variable-scale data structures. In *Proceedings of 14th ICA/ISPRS Workshop on Generalisation and Multiple Representation*, pages 1–9.

Meijers, M. (2011b). *Variable-scale Geo-information*. TU Delft, Delft University of Technology.

Meijers, M., van Oosterom, P., and Quak, W. (2009). A storage and transfer efficient data structure for variable scale vector data. In *Advances in GIScience*, pages 345–367. Springer Science Business Media.

Molenaar, M. and van Oosterom, P. (2009). Conceptual tools for specifying spatial object representations. In Madden, M., editor, *Manual of Geographic Information Systems/ed*, pages 47–69. Bethesda: American Society for Photogrammetry and Remote (ASPRS).

Rigaux, P., Scholl, M., and Voisard, A. (2001). *Spatial Databases: With Application to GIS (The Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann, 1 edition.

Sugumaran, R. and Degroote, J. (2010). *Spatial Decision Support Systems: Principles and Practices*. CRC Press, 1 edition.

van Oosterom, P. (1999). Spatial access methods. In Goodchild, M. F., Longley, P. A., Maguire, D. J., and Rhind, D. W., editors, *Geographical Information Systems Principles, Technical Issues, Management Issues, and Applications*, volume 1. John Wiley & Sons.

van Oosterom, P., Martinez-Rubi, O., Ivanova, M., Horhammer, M., Geringer, D., Ravada, S., Tijssen, T., Kodde, M., and Gonçalves, R. (2015). Massive point cloud data management: Design, implementation and execution of a point cloud benchmark. *Computers & Graphics*, 49:92–125.

van Oosterom, P. and Meijers, M. (2013). Vario-scale data structures supporting smooth zoom and progressive transfer of 2d and 3d data. *International Journal of Geographical Information Science*, 28(3):455–478.

van Oosterom, P., Meijers, M., Stoter, J., and Šuba, R. (2014). Data structures for continuous generalisation: tGAP and SSC. In *Lecture Notes in Geoinformation and Cartography*, pages 83–117. Springer Science Business Media.

van Oosterom, P., Quak, W., Tijssen, T., and Verbree, E. (2000). The architecture of the geo-information infrastructure. In *Proceedings of UDMS 2000, 22nd Urban Data Management Symposium, Delft, September 11-15, 2000*. Urban Data Management Society.

van Oosterom, P. and Stoter, J. (2010). 5D data modelling: Full integration of 2D/3D space, time and scale dimensions. In *Geographic Information Science*, pages 310–324. Springer Science Business Media.

Worboys, M. and Duckham, M. (2004). *GIS: A Computing Perspective, Second Edition*. CRC Press, 2 edition.

A INITIAL CONCEPTUAL DESIGN