



UNIVERSITEIT VAN PRETORIA  
UNIVERSITY OF PRETORIA  
YUNIBESITHI YA PRETORIA

# **Orchestrating standard web services to produce thematic maps in a geoportal of a spatial data infrastructure**

by

Victoria-Justine Rautenbach

Submitted in partial fulfilment of the requirements of the degree

**Magister Scientiae (Geoinformatics)**

**in the Faculty of Natural & Agricultural Sciences**

**University of Pretoria**

**Pretoria**

18 January 2013

## Abstract

Cartography is the science and art of making maps and thematic cartography is a subsection that deals with the production of thematic maps. A thematic map portrays the distribution of features, incidents or classifications related to a specific topic. With the rapidly increasing volumes of data, thematic maps allow users to efficiently analyse data and identify trends quicker. A spatial data infrastructure (SDI) focuses on making data available and ensures data interoperability through a geoportal and associated web services for discovery, display, editing, and analysis. Implementations of web service standards by the Open Geospatial Consortium (OGC), and the ISO/TC211, *Geographic information/Geomatics* enable the display, query and custom visualisation of spatial data in a geoportal. In the past, sophisticated cartographic methods have been mainly available on desktop applications, but with the advances in web mapping technology these methods have become increasingly popular on the Web. Currently, producing thematic maps using web services is a manual process that requires quite a lot of custom programming. The orchestrations of standard web services automate the process to produce thematic maps in a geoportal. It is preferable to use standard web services as opposed to customised programming; the standards provide flexibility, interoperability, and standard protocols, to name a few benefits. The goal of this research was to determine how standard OGC web services could be orchestrated to produce thematic maps within the geoportal of an SDI. To achieve this goal, an orchestrated thematic web service, named ThematicWS, was constructed from existing implementations of individual standard OGC web services, which are monolithic and interchangeable. The thematic cartographic process for producing choropleth and proportional symbol maps was investigated to model the process and obtain a set of steps. Experiments were performed to determine which existing web service standards could be used in the process. ThematicWS was developed using existing implementations of the following standards: WFS to retrieve the attribute data, WPS for the wrapping of custom functionalities (statistical processing and SLD generation), and a WMS to produce the thematic map image. The 52° North and ZOO project frameworks' orchestration capabilities were evaluated for to determine the suitability for producing thematic maps. The evaluation showed that orchestration is possible in both frameworks. However, there are limitations in both frameworks for automatic orchestration such as the lack of semantic information and poor usability of the framework. The use of WPS services to wrap custom functionalities and to provide a standard interface has proved to be useful for the orchestration of standard web services. ThematicWS was successfully implemented based on standard web service implementations using both workflow scripting and workflow modelling. The orchestrated ThematicWS can be called and consumed by a geoportal of an SDI to produce thematic maps according to user defined parameters.

## Acknowledgements

Firstly, I would like to thank my supervisor, *Dr Serena Coetzee*, without whom this research would not have been possible. It has been a great honour and pleasure to work with Dr Coetzee. She inspires hard work and excellence from those she works with, and is always willing to brainstorm and provide much needed advice. I would like to take this opportunity to thank Dr Coetzee for all the guidance, advice, and the once-in-a-lifetime opportunities. It has been a great privilege, thank you!

All the members of the South African/Poland bilateral project entitled, *Volunteered Geographical Information (VGI) for Spatial Data Infrastructures (SDIs) and Geoportals*, especially my Polish colleagues *Professor Adam Iwaniak* and *Mr Marek Strzelecki*.

The financial assistance of the *National Research Foundation (NRF)* and *Department of Science and Technology (DST)* towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attributed to the *National Research Foundation* and *Department of Science and Technology*.

Lastly, I would like to thank all *my family, friends, and colleagues* for their direct or indirect support in this process during the last two years. I am very blessed to have such wonderful and amazing people around me. Thank you all.

## Table of Contents

<b>Abstract .....</b>	<b>ii</b>
<b>Acknowledgements .....</b>	<b>iii</b>
<b>Chapter 1 Introduction .....</b>	<b>1</b>
1.1. Overview .....	1
1.2. Problem statement.....	2
1.3. Objectives .....	3
1.4. Methodology .....	3
1.5. Significance of the research .....	4
1.6. Publications from this research.....	4
1.7. Overview of the sections in this dissertation.....	5
1.8. Contributions to scientific research.....	6
<b>Chapter 2 Thematic cartography .....</b>	<b>7</b>
2.1. Introduction .....	7
2.2. Overview .....	7
2.3. Map design cycle.....	8
2.4. Symbolisation .....	11
2.5. Standardisation and classification methods.....	13
2.6. Overview of thematic map types.....	16
2.7. Theoretical thematic cartography .....	19
<b>Chapter 3 Web services, OGC standards and Orchestration .....</b>	<b>23</b>
3.1. Introduction .....	23
3.2. General web services .....	23
3.3. OGC web services.....	25
3.4. Orchestration of web services .....	32
3.5. Styled Layer Descriptor (SLD).....	35
3.6. Symbol Encoding (SE) and Filter Encoding (FE).....	44
3.7. Common Query Language (CQL).....	45
<b>Chapter 4 Spatial data infrastructure and Geoportals .....</b>	<b>46</b>
4.1. Introduction .....	46
4.2. Spatial Data Infrastructure (SDI).....	46

4.3. Geoportals .....	49
4.4. Moving towards an intelligent geoportal .....	50
<b>Chapter 5 Related work .....</b>	<b>51</b>
5.1. Introduction .....	51
5.2. SLD experiments and implementations .....	51
5.3. WPS research.....	55
5.4. Web service implementations and orchestration .....	55
5.5. Semantic web services .....	57
<b>Chapter 6 Modelling the thematic cartography process .....</b>	<b>58</b>
6.1. Introduction .....	58
6.2. Evaluation of thematic cartography implementations .....	58
6.3. Model of thematic cartography process .....	78
6.4. Conclusion .....	82
<b>Chapter 7 Design and implementation of ThematicWS .....</b>	<b>83</b>
7.1 Introduction .....	83
7.2 Functional requirements .....	83
7.3 Non-functional requirements.....	85
7.4 Architectural design .....	86
7.5 Detailed design .....	90
7.6 Implementation of ThematicWS service .....	98
<b>Chapter 8 Results of an evaluation of the orchestration capabilities of ZOO project and the 52° North framework for an intelligent geoportal .....</b>	<b>101</b>
8.1. Introduction .....	101
8.2. Overview of the WPS standard.....	103
8.3. Overview of WPS frameworks .....	104
8.4. Evaluation and results .....	105
8.5. Discussion of results .....	108
8.6. Conclusion .....	110
<b>Chapter 9 Discussion of results .....</b>	<b>111</b>
<b>Chapter 10 Conclusion .....</b>	<b>114</b>
10.1. Introduction .....	114
10.2. Main results from this dissertation .....	114

10.3. Recommendations for further research .....	115
<b>References .....</b>	<b>116</b>
<b>Referenced standards .....</b>	<b>124</b>
<b>Referenced websites .....</b>	<b>126</b>
<b>Appendix A. Acronyms and abbreviations.....</b>	<b>127</b>
<b>Appendix B. OGC web service standards request parameters.....</b>	<b>129</b>
<b>Appendix C. Examples of thematic maps produced by the ThematicWS service.....</b>	<b>132</b>
<b>Appendix D. ThematicWS functions .....</b>	<b>138</b>
D.1 Statistical processing service functions.....	138
D.2 SLD style sheet preparation service functions .....	141
<b>Appendix E. Journal publication and referenced conference paper .....</b>	<b>143</b>
E.1 Results of an evaluation of the orchestration capabilities of the ZOO project and the 52° North framework for an intelligent geoportal .....	143
E.2 Orchestrating OGC web services to produce thematic maps in a spatial information infrastructure .....	144

## List of Tables

Table 1. Summary of classification methods (Slocum et al., 2009) .....	16
Table 2. Comparison between the different types of thematic maps (Rita et al., 2010) .....	41
Table 3. Configuration options of the thematic engine (Sandvik, 2008) .....	65
Table 4. Parameters needed to create a chart with Google Chart API.....	74
Table 5. Thematic cartography methods available in the evaluated implementations.....	76
Table 6. Classification methods available in the evaluated implementations .....	76
Table 7. Data standardisation available in the evaluated implementations .....	77
Table 8. Visualisation and symbology available in the evaluated implementations.....	78
Table 9. Step in thematic cartography process with corresponding OGC web service .....	88
Table 10. The parameters of a GetThematicMap request .....	98
Table 11. Comparison of 52° North and ZOO Project framework according to the criteria specified .....	106
Table 12. List of abbreviations used in this dissertation .....	127
Table 13. The parameters of a GetFeature request (OGC 2010b).....	129
Table 14. Parts of Execute operation request.....	130
Table 15. The parameters of a GetMap request (OGC 2003) .....	131

## List of Figures

Figure 1. Flow diagram depicting the map design cycle .....	10
Figure 2. Interface of the TypeBrewer application .....	11
Figure 3. Illustrating the ease of understanding between symbols and text on a map .....	12
Figure 4. An example depicting the difference between classed and unclassed maps .....	12
Figure 5. Classification of geographic data (Tyner 2010) .....	13
Figure 6. The above maps are of the Gauteng province in South Africa depicting the population of each municipality .....	18
Figure 7. Examples of the Isarithmic, Dasymetric map, and Flow map from Slocum et al. (2010).....	19
Figure 8. Home screen of the ColorBrewer website .....	21
Figure 9. Communication between web service and client.....	24
Figure 10. The OGC Web Services architecture (OGC 2002).....	27
Figure 11. Basic sequence diagram depicting the different requests and responses of the WMS .....	29
Figure 13. Basic sequence diagram depicting the different requests and responses of the WFS.....	30
Figure 14. Basic sequence diagram depicting the different requests and responses of the WPS .....	31
Figure 15. SLD code to produce a choropleth map with three classes, adapted from the SLD Cook book (Pumphrey, 2010) .....	38
Figure 16. The resulting thematic map of the Gauteng municipalities based on population.....	38
Figure 17. SLD code to produce a proportional symbol map with three classes, adapted from the SLD Cook book (Pumphrey, 2010) .....	40
Figure 18. The resulting thematic map of the Gauteng municipalities based on population.....	40
Figure 19. Example of a style sheet implementation of the dynamic symbolizer.....	42
Figure 20. Example of style sheet implementing parameter substitution.....	43
Figure 21. An example of a diagram map produced with the GeoServer charts extension .....	44
Figure 22. SDI hierarchy (Rajabifard et al., 2006, World Bank, 2011).....	47
Figure 23. Classification of web portals .....	49
Figure 24. KML code use for thematic mapping, from the KML for Thematic Mapping (Sandvik, 2009)...	53



Figure 25. 3-Tier system architecture as well as the process of the thematic map service (Hong & Lin, 2005).....	54
Figure 26. Basic Process of the XSLT Template approach (Čerba, 2010).....	54
Figure 27. ArcMap screen to produce an unclassified choropleth map.....	59
Figure 28. ArcMap screen to produce a classified choropleth map.....	60
Figure 29. Histogram to evaluate the different classification methods.....	60
Figure 30. ArcMap screen to produce a proportional symbol map .....	61
Figure 31. Examples of thematic maps generated with ArcGIS .....	61
Figure 32. QGIS user interface for producing a classified choropleth map.....	62
Figure 33. QGIS user interface for producing proportional symbol maps.....	63
Figure 34. GeoDa, the produced choropleth map and the different types of classification can be seen ...	64
Figure 35. GeoDa dialog for selecting the attribute and number of classes .....	64
Figure 36. Code to generate a choropleth map .....	66
Figure 37. Code to generate a proportional symbol map.....	66
Figure 38. Web interface implementing the Thematic Mapping API.....	67
Figure 39. A bar chart showing the population per country produced with the Thematic API, displayed in Google Earth.....	68
Figure 40. indiemapper interface to select spatial data .....	69
Figure 41. Interface for selecting the thematic mapping type and attribute to be mapped .....	69
Figure 42. indiemapper with styling options and produced choropleth map.....	69
Figure 43. indiemapper with styling options and produced proportional symbol map .....	70
Figure 44. Interface for styling and the produced choropleth thematic map.....	71
Figure 45. Interface for styling and the produced proportional symbol thematic map .....	71
Figure 46. A class break renderer choropleth map example. Courtesy of ESRI Resource Centre .....	72
Figure 47. A class break renderer choropleth map example. Courtesy of ESRI Resource Centre .....	73
Figure 48. A unique value renderer example, courtesy of ESRI Resource Centre.....	73

Figure 49. Animated map example, courtesy of ESRI Resource Centre .....	74
Figure 50. Outline of the thematic cartography model .....	79
Figure 51. Flow diagram depicting the thematic cartography process .....	80
Figure 52. Simplified step of ThematicWS .....	87
Figure 53. Use case diagram depicting the functions of the ThematicWS service .....	89
Figure 54. Sequence diagram illustrating the orchestration of web services in ThematicWS .....	89
Figure 55. Use case diagram depicting the functions of statistical processing web service .....	92
Figure 56. Process in the WPS for statistical processing .....	93
Figure 57. Use case diagram depicting the functions of the SLD generator web service .....	94
Figure 58. Process in the WPS for SLD generator .....	95
Figure 59. Sequence diagram illustrating the orchestration of web services in ThematicWS with a workflow modeller .....	99
Figure 60. Depicts the architecture of the 52° North framework .....	104
Figure 61. Depicts the architecture of the ZOO framework, adapted from ZOO Project .....	105
Figure 62. A sequence diagram depicting the orchestration of web services to create thematic maps ..	110
Figure 63. Current SDI geoportals with available service types, adapted from the INSPIRE Network Service Architecture (2007).....	112
Figure 64. Components of the ThematicWS (Rautenbach et al., 2012a) .....	112
Figure 65. Request for producing an equal interval choropleth thematic map with 6 classes .....	132
Figure 66. A classed choropleth map (equal interval classification with 6 classes) of the population of the United States of America .....	132
Figure 67. Request for producing an equal interval proportional symbol thematic map with 6 classes..	133
Figure 68. A classed proportional symbol map (equal interval classification with 6 classes) of the population of the United States of America.....	133
Figure 69. Request for producing a quantile interval choropleth thematic map with 6 classes .....	134
Figure 70. A classed choropleth map (quantile classification with 6 classes) of the population of the United States of America .....	134
Figure 71. Request for producing a quantile interval proportional symbol thematic map with 6 classes.	135

Figure 72. A classed proportional symbol map (quantile classification with 6 classes) of the population of the United States of America ..... 135

Figure 73. Request for producing an unclassed choropleth thematic map ..... 136

Figure 74. An unclassed choropleth map of the population of the United States of America ..... 136

Figure 75. Request for producing an unclassed choropleth thematic map ..... 137

Figure 76. An unclassed proportional symbol map of the population of the United States of America ... 137

# Chapter 1 Introduction

## 1.1. Overview

At present, data is collected throughout our entire day, for example, through GPS navigation and social networks. The problem with this large amount of data being collected is how to present it; this provides an opportunity to create amazing interfaces for the presentation of the data. An unknown author stated that ‘the 19<sup>th</sup> century culture was defined by the novel, 20<sup>th</sup> century by cinema, the culture of the 21<sup>st</sup> century will be defined by the interface’. An interface can be a powerful narrative device, the way data is present can influence the meaning of the data and the message it conveys (Koblin, 2011). Tim O’Reilly has been quoted saying that people think of data visualisation as output, but that data visualization is becoming a means of input and control. Data visualization would then become an interface rather than a report, with the ability to manipulate the data in real time allowing the user to ensure the correct message is conveyed.

Visualisation interfaces can be a very powerful tool for spatial data, which can aid users with information visualisation and data exploration. A thematic map is an example of information visualisation; a specific type of map that is designed to communicate information about a single topic or theme (Wade & Sommer, 2006; Tyner, 2010). Thematic maps portray the geographic distribution of features, incidents or classifications related to a specific topic, making it easier for the user to identify patterns within the datasets (Parr, 2000; Robinson et al., 1995).

In 2005, only 15% of the earth’s surface was mapped to a geocodable level of detail. However, since then there have been several very successful initiatives and as a result in 2009 it was reported that only 17% was still unmapped (Katragadda, 2009). These mapping initiatives produced huge amounts of data ranging from satellite images to vector layers. Spatial data infrastructures (SDI) emerged as a result of the ever-expanding volume of spatial data and the need to share this data, to name a few. An SDI makes it possible for different users to access spatial data and other relevant datasets. In an SDI, a geoportal is typically used to provide access to the spatial *data* and associated web services for discovery, display, editing and analysis (Maguire & Longley, 2005).

There have been numerous successful SDI implementations across the world, for example, the European SDI, **I**nfrastructure for **S**Patial **I**nfo**R**mation in **E**urope (INSPIRE) and the Canadian Geospatial Data Infrastructure (CGDI). An SDI is a combination of technology, policies, standards, human resources and related activities essential to acquire process, distribute, use, maintain and preserve spatial data throughout all levels of government, private sectors and academia (Coetzee, 2008). The collaboration of multiple organisations reduces the risk for one organisation; partners or stakeholders to share in responsibilities and resources. The focus of an SDI is on data harmonization and making *data* available through the geoportal with associated web services for discovery, display, editing and analysis.

An *intelligent geoportal* is a geoportal that provides access to information rather than data, through complex functionality exposed as a simple user interface for a user in a specific application domain (Iwaniak et al., 2011). An intelligent geoportal requires more advanced and intelligent web services that employ cartographic methods for information presentation, instead of data to the user. The intelligence implied in the term 'intelligent geoportal' could be provided using either semantics or automation of process. One method of automation is the orchestration of web services to create a new, more complex web service to perform a specific process.

Interoperability of web services relies on the development and use of uniform standards. The two main standardisation organisations in the geographic information society is the International Organization for Standardisation (ISO) and the Open Geospatial Consortium (OGC), which are widely accepted and implemented in SDIs such as INSPIRE. The OGC web services are generally used to implement geospatial web services in geoportal for discovery, display, editing and analysis.

Orchestration is the automated arrangement of web services in a predefined pattern based on local decisions about their interactions with one another at the message and/or execution level (Sun et al., 2010; Suazo & Aguirre, 2005). Orchestration enables sophisticated forms of interaction with inherited intelligence or implicit automatic control. Complex web services can be created to perform specific tasks or processes such as producing thematic maps. The goal of this project is to determine how standard OGC web services can be orchestrated to create a new orchestrated web service to produce thematic maps within the geoportal of an SDI.

## **1.2. Problem statement**

Advanced cartographic methods are commonly available on desktop applications; this is made possible through the technological improvements enabling users to process large datasets, and produce maps that could not be produced previously. Desktop applications are, however, not an optimal solution for data in an SDI, since the data can be distributed over several sources, leaving online processing the best option.

Web mapping technologies have rapidly evolved since the mid-nineties, with the development of web services and OGC web standards for example, and especially in the Web 2.0 era (Fu & Sun, 2010). Web mapping has proved to be useful for small businesses or government organisations that do not have the capital to invest in large-scale GIS software and required hardware. Cloud computing, which can be 'rented' when required, eliminates the need for expensive computing and storage facilities (World Bank, 2011).

Thematic maps allow users to analyse data in several ways, through identifying pattern and classification of data, for example. A thematic web service allows users to specify a link to the data source, enabling users to produce thematic maps without downloading the data.

A couple of thematic map types can be created with the use of OGC standards, such as the Web Map Service and Styled Layer Descriptor. However, some of the processing to create a thematic map needs to be done manually by the user, for example, classifying the data and creating an appropriate style sheet that can be time consuming. Orchestrating standard OGC web services could automate some of the manual labour required for producing thematic maps and have additional advantages, for instance, different OGC web service implementations can be exchanged (interoperability), existing OGC web service implementations can be re-used, saving on costs (code re-use), and well-known standardised OGC protocols are used.

### **1.3. Objectives**

The goal of this project was to investigate how standard OGC web services can be orchestrated in order to produce thematic maps. To achieve this goal, an orchestrated thematic web service, named ThematicWS, was constructed from individual standard OGC web services, which are monolithic and interchangeable.

The following are the secondary objectives of this project to achieve the main goal:

1. Perform a literature review of existing theory and related work.
2. Model a process for choropleth and proportional symbol thematic cartography methods.
3. Design ThematicWS, a web service that orchestrates OGC web services to produce thematic maps in a geoportal.
4. Evaluate the workflow modeller implemented such as 52° North and ZOO project's framework, in order to select framework for the implementation of ThematicWS.
5. Implement and evaluate a prototype of the ThematicWS service.

### **1.4. Methodology**

The empirical research methodology was used for this research. According to Olivier (2004), this methodology may be used to do exploratory research or to extend a theory. The method was chosen since this research project is an exploratory research project that investigates how standard web services can be orchestrated to develop a thematic web service to produce thematic maps.

The primary method of this research was prototyping and the secondary method a literature survey.

The literature survey provides the required background into the disciplines of cartography and computer science. This aided the author in understanding all aspects involved: thematic cartography, OGC standards and web services, spatial data infrastructure, spatial information infrastructure, geoportals, and other related work. A prototype of the orchestrated thematic web service was developed as a proof of concept. The last step of the project was to write this dissertation, which

combines my understanding of the relevant theory, previous research and the results of my empirical research and prototype of the thematic web service.

The ThematicWS service was only implemented to produce choropleth and proportional symbol maps. The main reason why these thematic map types were chosen is that in the evaluation done in Section 6.2 of this dissertation these map types were found to be the most popular types. This research focused on developing a new orchestrated thematic web service, rather than extending the styled layer descriptor (SLD) standard for visualisation of spatial data.

## 1.5. Significance of the research

The ability to produce thematic maps from distributed data with the use of a web service is very useful for decision makers in an SDI, to aid them in providing specific information about a particular location, general information about spatial patterns, and to compare patterns on two or more maps (Slocum et al., 2009). The current trend is to move from a data centric approach to a distributed approach, which will affect the manner in which we process our data.

At present, a couple of thematic maps can be created using OGC standards; however, the user has to drive the process. Currently, data processing and SLD generation need to be manually performed by the user, which can be very time consuming and often a repetitive process, in order to produce the desired thematic map. The development of an orchestrated thematic web service, ThematicWS, eliminated some of the manual work required in producing thematic maps and enabled better interoperability and re-use since the service was developed with OGC standards.

With the automation of the thematic cartography process, user interaction cannot be completely eliminated, especially for the verification of the map; a certain degree of automation has been achieved through the use of the ThematicWS service. Some level of human error is removed with the use of web services. For example, the statistical engine, which may eliminate some problems in that it has the ability to compute large datasets without any limitations. The use of web services also allows for code reuse, which has numerous advantages such as cost reduction.

## 1.6. Publications from this research

The following publications were a result of this research:

1. **Rautenbach, V.** (2011). 'n Webdiens vir tematiese kartering [A web service for thematic mapping]. *Afrikaanse Studentesimposium*, Pretoria, South Africa, 27 - 28 October 2011. (In Afrikaans).
2. **Rautenbach, V., & Coetzee, S.** (2011). Producing thematic maps using ISO and OGC standards. *International Organization for Standardization (ISO) Standards in Action Workshop*, Centurion, South Africa, 11 November 2011.

3. **Rautenbach V**, Coetzee S, and Iwaniak A (2013). Orchestrating OGC web services to produce thematic maps in a spatial information infrastructure, *Computers Environment and Urban Systems*, ISSN 0198-9715, **37**:107 - 120, DOI: [10.1016/j.compenvurbsys.2012.08.001](https://doi.org/10.1016/j.compenvurbsys.2012.08.001).
4. **Rautenbach, V.**, Coetzee, S., Strzelecki, M. & Iwaniak, A. (2012). Results of an evaluation of the orchestration capabilities of the ZOO project and the 52° North framework for an intelligent geoportal, *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Volume I-4, 2012 XXII ISPRS Congress, 25 August – 01 September 2012, Melbourne, Australia.

## 1.7. Overview of the sections in this dissertation

This research spans two disciplines: Geographic Information Science (GISc) and Computer Science. Thematic cartography, thematic mapping, spatial data infrastructure, and geoportals relate to geographic information science. The web service standards and the orchestration of OGC web services are computer science aspects covered in the research. Due to the cross-disciplinary nature of the project, an extensive literature review in both disciplines were required; the results of the literature review are presented in chapter 2 to 5, followed by research performed in chapter 6 to 10 to achieve the objectives set out in this project.

The remaining chapters of this dissertation are structured as follows:

**Chapter 2 – Thematic cartography.** The theoretical cartographic process based on current scientific literature is described in this chapter, together with other thematic mapping aspects such as map design and symbolisation.

**Chapter 3 – Web services, OGC standards and orchestration.** An overview of general web services, the relevant OGC web services, orchestration of web services and the technologies used to orchestrate web services, such as workflow modellers, are discussed.

**Chapter 4 – Spatial data infrastructure and geoportals.** Spatial data infrastructures, geoportals and the need for intelligent geoportals to provide users with information, e.g. thematic maps rather than data is explained.

**Chapter 5 – Related work.** An overview of related work in the fields of thematic mapping using OGC standards, WPS implementations and issues, and the orchestration of OGC web services is provided. Current challenges for the development of a ThematicWS are discussed.

**Chapter 6 – Modelling the thematic cartography process.** An evaluation of theoretical cartography process and thematic mapping implementations are described. Based on the evaluation a model of the thematic cartography process is produced.

**Chapter 7 – Design and implementation of the orchestrated thematic web service.** The technical requirements, non-functional requirements, architecture and the detailed design of ThematicWS, an



orchestrated web service, is presented. The implementation of the orchestrated web service is described, providing an overview of the technologies used to implement the custom services required for ThematicWS.

**Chapter 8 – Results of an evaluation of the orchestrated capabilities of ZOO project and the 52° North framework for an intelligent geoportal.** An evaluation of two orchestration platforms is given in this chapter, to assist in the selection of a platform for the development of the custom WPS services and selecting a workflow modeller to orchestrate services in order to create the ThematicWS service.

**Chapter 9 – Discussion of results.** The results obtained from the experiments and implementation of ThematicWS are discussed.

**Chapter 10 – Conclusion.** The most significant results obtained from this project and future research topics are discussed.

## 1.8. Contributions to scientific research

The contributions from this research towards the two disciplines of Geographic Information Science and Computer Science are listed in the following paragraphs:

**Model of the thematic cartography process.** The model developed of the thematic cartographic process for choropleth and proportional symbol maps is presented in Chapter 6. A flowchart is used to depict the model which, when combined with best practices, can be used for the implementation of expert systems and automatisisation of the thematic cartography process.

Theoretical thematic cartography is explained by many authors as separate components (map design, symbology, classification, etc.) forming the thematic process, but has rarely been combined in this fashion to demonstrate a sequence of steps to produce thematic maps.

**Evaluation of the orchestration capabilities of ZOO project and the 52° North framework.** The evaluation of orchestration capabilities in Chapter 8 provides a guide for developers attempting to orchestrate OGC web services. Criteria used for the evaluations can be applied to benchmark other frameworks' capabilities.

**Implementation of ThematicWS and the discussion of results.** The design and implementation of ThematicWS discussed in Chapter 7. ThematicWS was implemented using the service oriented architecture (SOA); technical and non-technical requirements are described in order to develop the detail design of the web service. The model developed in Chapter 6 was used to identify the different web service required for producing thematic maps. The detail of the implemented functions for the custom services is discussed in Appendix C.

## Chapter 2 Thematic cartography

### 2.1. Introduction

This chapter discusses the theoretical cartographic process for producing thematic maps based on current scientific literature. The choropleth and proportional symbol thematic maps are described in detail to provide the required cartographic background. Other thematic mapping aspects are also discussed such as the map design process and symbology.

### 2.2. Overview

Cartography is the science and art of making maps. This term encapsulates all the processes and steps involved in the production of all types of maps. A thematic map is a specific type of map that is designed to communicate information about a single topic or theme, such as population density or geology. Base data, in addition, is served to provide a spatial background or framework to aid in locating the distribution being mapped (Wade & Sommer, 2006; Tyner, 2010). Thematic maps portray the geographic distribution of features, incidents or classifications related to a specific topic (Parr, 2000; Robinson et al., 1995). Thematic cartography is a subsection of cartography that deals in detail with the production of thematic map. Thematic maps are usually targeted towards a specific audience. In recent years, the quantity of geographic data has increased rapidly, and to be able to use and interpret this data, thematic cartography has become ever more important.

Thematic maps are used to represent spatial or physical phenomenon (Sandvik, 2008) and the statistical information associated with the particular geographic area (Rita et al., 2010). Thematic maps emphasize the patterns that occur within a phenomenon or else one or more geographic attributes, such as population density (Slocum et al., 2009). Thematic maps can be used in a wide variety of ways. They can provide specific information about a particular location, general information about a spatial pattern and assist in the comparing of patterns on different maps but at the same location.

One of the more popular thematic map types is the Choropleth map. The choropleth map is, however, just one type available. There are numerous other types: proportional symbols map, dot density map, isarithmic map, dasymetric map, cartogram, and flow generalization map to name a few.

The type of thematic map selected to represent the information can contribute and aid in the interpretation of the map (Rita et al., 2010). Depending on how the information is presented, different patterns can be distinguished in the data. Within these types there are three different sub-types, namely classed, unclassed and animated maps (Slocum et al., 2009). Classed maps refers to maps where the data is grouped into classes of similar values. Unclassed maps is when values are proportionally assigned to groups. This type of map is a more accurate view of the real world. An

animated map is a particular representative of the capability of modern computer technology demonstrating a phenomenon changing over a period of time, for example, changing clouds.

A thematic map is one of the most powerful tools in an SDSS. It provides a visually communicated map that can aid in the decision-making process of a wide range of industries from the business to government sector. In these sectors mass amounts of data is stored in an SDI or Data Warehouse which is a distributed storage for spatial and non-spatial data (Iosifescu-Enescu et al., 2007). For this reason, the designing and implementing of an on-demand thematic map generation is needed. Thematic maps can be easily generated using a wide range of desktop applications and some web applications; however, no web service is available. One may ask, 'Why is a web service needed?' The answer is simple; the downloading of the data may be costly and the processing needed by the thematic engine may not be available to the user, thus a thematic web mapping service will make producing thematic maps more efficient for the users. At present, the Open Geospatial Consortium (OGC) has no explicit feature or standard for producing thematic maps (Rita et al., 2010).

With the growing popularity of home computers and geographic information system (GIS) application, anybody can produce a map (Slocum et al., 2009). This is referred to as the democratization of cartography. The production of maps with these software and use of wizards does not guarantee that the resulting map is well designed or accurate. At present, there is no expert system for thematic mapping (a computer that automatically makes decisions about symbolisation and classification).

### **2.3. Map design cycle**

Map design choices involve a combination of intuition and rational choice. Slocum et al. (2009) describes cartographic design as a mental and physical process in which maps are conceived and created. Maps do not necessarily communicate knowledge, but rather provide information that stimulates and suggests knowledge. Tyner (2010) states that the mapping process is not a linear process, but is represented in this manner to simplify the representation of the process. The design process has two main goals (Slocum et al., 2009): to produce a map that appropriately serves the map's intended use, and to communicate the map's information in the most efficient manner simply and clearly.

Map reading is a task that requires a certain amount of skill; however, map design is equally important. The design of the map must be such that a person of low skill will be able to read the map. The following must be considered: deciding on the best presentation of data on a thematic map requires the cartographer to have a critical view of the data to be mapped, as well as the symbolisation that will be used to represent the data (Cuff & Mattson, 1982). This is necessary since the symbolisation that is used is very important. If the symbol is incorrectly used, the meaning of the map may be altered. Symbology is discussed in Section 2.4.

A naïve mapmaker will most likely not consider the optimal layout for communicating the information on the map; they are more likely to just use the available data and options. For this reason the following issues are important to consider when producing thematic maps (Slocum et al., 2009):

1. Consider what the real-world distribution of the phenomena might look like.
2. The purpose of the map and its intended audience.
3. Is the collected data appropriate for the purpose of the map?
4. Ensuring that the map is useful and informative.

In the design process, decisions need to be made about all the map elements: frame line and neat line, mapped area, inset, title and subtitle, legend, data source, scale, and orientation. Cartographers have over the years developed guidelines and basic principles in order to produce the best quality map (Slocum et al., 2009; Tyner, 2010; Dent, 1996).

Robinson et al. (1995) divided the design process into the following three steps:

1. Graphic ideation:

In this step the cartographer decides on the type of map, spatial format, basic layout of the map, which data will be represented, and mapping technique used. This results in a design plan for the map.

2. Graphic plan:

Alternatives are analysed and compared with the current plan's limits. The cartographer also has to decide on the kind of symbology, number of classes and class limits, colours to be used, topographical relationship, and general line weights, to name a few. In this stage all decisions should be made except the minor choices.

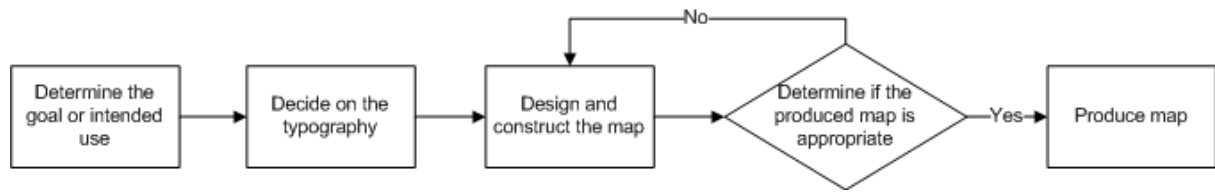
3. Detailed specifications:

The cartographer should define all symbols, line weights, screens, colours, letter sizes, etc.

Slocum et al. (2009) defines the steps in the design process as follows:

1. Consider what the real-world distribution of the phenomenon would look like.
2. Determine the purpose of the maps, and the intended audience.
3. Collect data appropriate for the map's purpose.
4. Design and construct the map.
5. Determine whether users find the map useful and informative; if not, repeat number 4.

Tyner (2010) divides the mapping process into the following categories: planning, data analysis, presentation, critique and editing, and lastly, production. Similar to these, Dent (1996) identified the following stages: problem identification, preliminary ideas, design refinement, analysis, decision, and implementation.



**Figure 1. Flow diagram depicting the map design cycle**

What all the design processes above have in common is that, in the last phase, the map must be evaluated in order to determine the success of the map. Dent (1996) provides the following characteristics of a successful map:

1. A map should satisfy the needs of the users.
2. A map should be easy to use.
3. Maps should be accurate, present information without adding error or distortion, and avoid misinterpretation.
4. A map should be clear, and good-looking.
5. Maps would ideally permit interaction, allowing the user to change, update or personalise the map.

TypeBrewer is a free tool that provides a semi-structured environment for mapmakers, which allows them to explore different typography (refer to Figure 2). TypeBrewer is not a mapmaking application, but rather a graphic designer, in which the users can explore alternative typography and see the effect of different elements on the overall look and feel of the map.

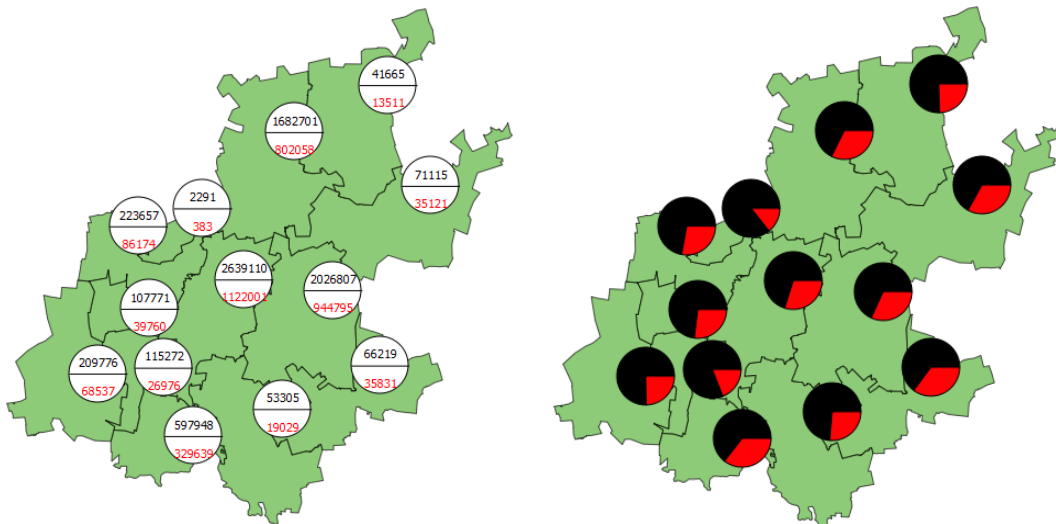


Figure 2. Interface of the TypeBrewer application

## 2.4. Symbolisation

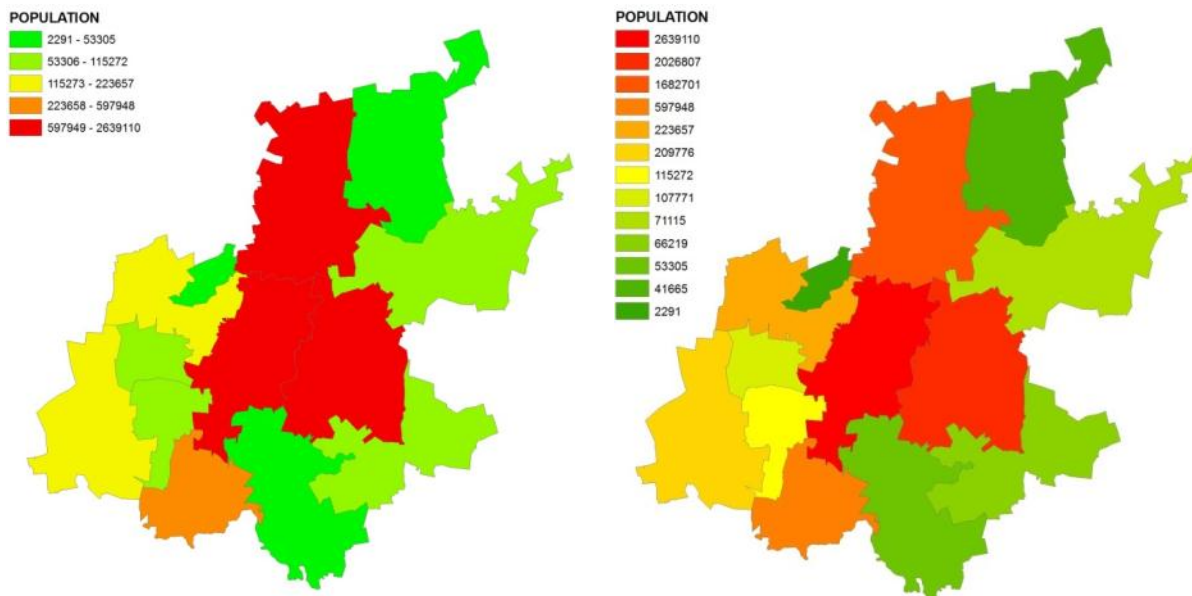
A map is a graphical form of communication in which symbols are primarily used to convey meaning. Tyner (2010) suggests that symbols can be seen as the graphic language of maps, the selection and design of the symbols being very important in the success of the map.

Thematic data is statistical in nature. Statistical data can be successfully displayed in a tabular form; however, thematic maps are used in order to represent and visualise the spatial component of this data. In tabular form statistical data can be very successful, but the spatial component is lost in the tabular form; on a thematic map the spatial patterns can be more easily identified and examined. So the question is: why don't we put the actual number on a map? The reason for this is that symbols are easier to understand and simplify the map; this is illustrated in Figure 3. If all the raw data were to be placed on the map it would be difficult to differentiate between different location values. Well-chosen symbols result in an easily understandable map of which the message is clear and unambiguous, conversely, poorly-chosen symbols lead to confusion and may mislead readers. Thus the selection of symbols is very important in order to achieve the goal set out for the map.



**Figure 3. Illustrating the ease of understanding between symbols and text on a map**

The human eye has a limited ability to distinguish between large numbers of colours and symbols (Stern et al., 2006). This makes an unclassed map with a large number of values difficult to read. In these cases, a classed map can be very useful, especially to portray quantitative data as a thematic map. In this process, a smaller number of data classes and colours or symbols are chosen to distinguish more easily between them. Figure 4 shows the difference between the same dataset being portrayed as an unclassed and classed map. The dataset that was used here is small, but the difference is still very visible between the two techniques.



a) Classed choropleth map with five classes

b) Unclassed choropleth map

**Figure 4. An example depicting the difference between classed and unclassed maps**

Data can be defined into four levels of measurements: nominal, ordinal, interval and ratio classification. Nominal classification is qualitative information; the symbols selected for this type of data must represent the data, for example, a pictorial symbol can be used to represent the point where a fire was started in a park. Ordinal, interval, and ration classification are all quantitative information. Figure 5 summarises the different types of data and the different types of symbols that could be used to represent them on a map.










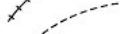



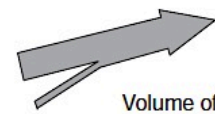






	Nominal Measurement	Ordinal Measurement	Interval/Ratio
POINT SYMBOLS	 Building  City  Mountain	 Village  Town  City	 Population 1,000,000 500,000 250,000
LINE SYMBOLS	 Road  Railroad  Trail	 Heavy-Duty Road  Medium-Duty Road  Light-Duty Road	 Volume of Traffic
AREA / VOLUME SYMBOLS	 Grassland  Orchard	 High-Density  Medium-Density  Low-Density	Population Per Sq. Mi.  100 25 0

Figure 5. Classification of geographic data (Tyner 2010)

## 2.5. Standardisation and classification methods

Standardisation of data is necessary for more meaningful thematic map presentations (Stern et al., 2006). We standardise data in order to make our data comparable to another, to show the ratio and enable a better analysis of the data. There are a number of standardisation methods available: ratio, density, rate and area-based standardisation. Ratio standardisation is the most common method that divides an area-based numerical dataset by another area-based numerical dataset. The numerator and denominator are of the same measurement unit, and the result can be expressed as a proportion. Density standardisation is used to indicate the density; division of a non-area based variable by an area is the basic process to perform this standardisation technique. Rate standardisation is used to compute the ratio of two non-area based variables; the result will always be in rates. Area-based



standardisation divided an area size by a non-area based variable. Standardisation is necessary when the user wants to compare other datasets to the original dataset.

Data classification is the process of dividing raw data into classes or groups, after which each class is represented by a unique symbol (Slocum et al., 2009). The result of the classification process is a classed map. If no classification method is performed on the data, each distinct value in the dataset is then depicted using a unique symbol.

Classification is used to structure the thematic map in such a way that the desired message is conveyed. A classed map has the following advantages: it makes it easier for the reader to discriminate among numerous different areal symbols and traditionally is very tedious to produce an unclassed map. The latter is not applicable anymore; with high performing GIS software today, an unclassed map is easily produced.

An important rule for all classification methods is that a single value may only appear in one class and the class limits shall not overlap. The most important consideration with regards to classification is the amount of classes the data should be divided into. The correct number of classes is between 4 and 8 and the standard is to use 5 or 6 classes (Stern et al., 2006). How the user chooses to portray the data on the map has a great influence on which classification method is selected.

### 2.5.1. Equal intervals

With the equal interval method, the range of the data is divided into equal interval sizes. The distribution of the data is not considered in the equal interval method, only the size of the class. The class width or interval can be computed easily with the following equation:

$$\frac{\text{Range}}{\text{nr. of Classes}} = \frac{\text{highest} - \text{lowest value}}{\text{nr. of Classes}}$$

Numerous advantages of the equal interval method are as follows: the method is easy to compute, map readers find it easy to interpret, and there are no missing values or gaps. The method, however, has one major drawback: the fact that it does not consider the distribution of the data.

### 2.5.2. Quantiles

The quantiles method first ranks the data and then places an equal number of observations within each class. Numerous identical data values can complicate this method since these data values should appear in the same class according to the classification rules. The method is very useful especially with ordinal data.

$$\text{number of observation in each class} = \frac{\text{Total Number of Observations}}{\text{nr. of Classes}}$$

The quantiles method has the following advantages: class limits can be computed easily and the percentage of observations in each class is equal. Similar to the equal interval method, this method fails to consider the distribution of the data. Another disadvantage is that there may be gaps between classes.

### **2.5.3. Mean-standard deviation**

This mean-standard deviation method is the first method thus far that takes the distribution of the data into consideration. Repeatedly adding or subtracting the standard deviation from the mean of the data forms classes. This method only works on normally distributed datasets. The method requires basic statistics; the mapmaker must thus be able to perform basic statistical functions in order to classify the data.

### **2.5.4. Maximum breaks**

The maximum breaks method is a simple method for creating classes when considering individual data values and grouping similar data values. The data is ordered ascending and the differences between adjacent values are computed. The largest differences collected will serve as the class breaks. The method only considers the largest breaks and may miss natural clusters that can occur within the dataset.

### **2.5.5. Natural breaks**

As mentioned, the maximum breaks fail to consider the natural clusters within the data. The natural breaks method is a solution to this problem. To determine the natural grouping of the data, graphs are examined to locate the logical breaks in the data. The purpose of the natural breaks method is thus to minimize differences between the data values in the same class and maximize the difference between the classes.

### **2.5.6. Optimal**

The optimal method addresses the limitations of the maximum break and natural breaks methods. The optimal method places similar data values in the same class by minimizing classification errors. An algorithm is used, such as the Jenks-Caspall and Fisher-Jenks algorithm to compute the classes. This method can be used if the user is not sure which classification method to use.

**Table 1. Summary of classification methods (Slocum et al., 2009)**

	<b>Equal Interval</b>	<b>Quantile</b>	<b>Mean Standard Deviation</b>	<b>Maximum Breaks</b>	<b>Natural Breaks</b>	<b>Optimal</b>
<b>Considers the distribution of the data</b>	Poor	Poor	Good	Good	Very Good	Very Good
<b>Ease of understanding the concept</b>	Very Good	Very Good	Very Good	Very Good	Good	Good
<b>Ease of computation</b>	Very Good	Very Good	Very Good	Very Good	Very Good	Very Good
<b>Ease of understanding legend</b>	Very Good	Poor	Good	Poor	Poor	Poor
<b>Legend values match data in class</b>	Poor	Very Good	Poor	Very Good	Very Good	Very Good
<b>Acceptable for ordinal data</b>	UA <sup>1</sup>	UA	UA	UA	UA	UA
<b>Assists in selecting number of classes</b>	Poor	Poor	Poor	Poor	Good	Very Good

<sup>1</sup>Unacceptable

## 2.6. Overview of thematic map types

Determining the different types of thematic maps seems like a trivial process; however, after investigation it was clear that some inconsistencies occur with regards to thematic map types. Dent (1996) divides thematic maps into two groups: qualitative and quantitative. Qualitative thematic maps are used to show the spatial distribution of the data, whereas quantitative thematic maps rather display the spatial aspects of numeric data. Quantitative maps listed by Dent are: choropleth, dot density, proportional symbol, isarithmic, cartogram, and flow maps. Slocum et al. (2009) lists the thematic map types as choropleth, Dasymetric, isarithmic, proportional symbol, dot, cartogram and flow mapping. Haklay (2010) extends this and lists the following types: dot, isolines, isoareas, density, choropleth, graduated symbol, diagram, cartogram and flow map. From these lists it can be seen that the naming and grouping of types differ. For example, Haklay refers to graduated symbols and Slocum et al. to proportional symbols; however, both these types refer to the same type of thematic map. Slocum et al. refers to graduated symbol maps as a sub type of the proportional symbol map in which the symbols are grouped into classes and a predefined symbol is used to represent the value of the attribute. Secondly, Slocum et al. groups isolines and isoareas into isarithmic mapping. From this it can be concluded that the naming and grouping of the thematic map types differ from sources and this may create confusion amongst users.

In this section, a brief overview of the following thematic map types are provided; choropleth, proportional symbol, dot density, cartogram, isarithmic, dasymetric and flow maps. The choropleth and proportional symbol techniques will be discussed in Section 2.7 in greater detail.

### **2.6.1. Choropleth map**

John Kirtland Wright coined the term *choropleth map* in 1938 (Crampton, 2009). Choropleth is the term that describes the most popular form of thematic map today. Choropleth maps are used to portray data collected within enumeration/pre-defined, such as country or state, boundaries by colouring or shading these areas. The choropleth map does not take into consideration variations within the enumeration unit and it is difficult to interpret for non-uniformed distributions. However, the choropleth map is still very popular; Sui and Holt suggest that this is due to the flattened learning curve with regards to spatial analysis (Sui & Holt, 2008).

### **2.6.2. Proportional symbols map**

The proportional symbol technique uses a symbol scaled in proportion to the magnitude of the data associated with a specific location or enumeration area. The location of the symbol on the map can either be at the true location or at a conceptual point within an enumeration unit. The data displayed with the proportional symbol map is usually raw data in contrary to the choropleth map, which requires standardised data.

### **2.6.3. Dot density map**

A dot density map is used to illustrate variations in spatial density. A dot can represent a single occurrence or multiple. The size, shape and colour of the dot does not change, but the frequency of dots changes from area to area in proportion to the number of objects being represented. This is an effective tool to show the overall pattern of distribution.

### **2.6.4. Cartogram**

A cartogram depicts specific statistical information in a graphic form. A cartogram does not represent geographic space or distance, it changes the size of objects depending on a certain attribute; some do not consider a cartogram a true map.

### **2.6.5. Isarithmic map**

An isarithmic map is used to represent continuous data with a third dimension such as elevation or precipitation. This type uses smooth lines created by interpolating a set of isolines between locations of known values. An isopleth map is a sub type in which the sample points are associated with enumeration units.

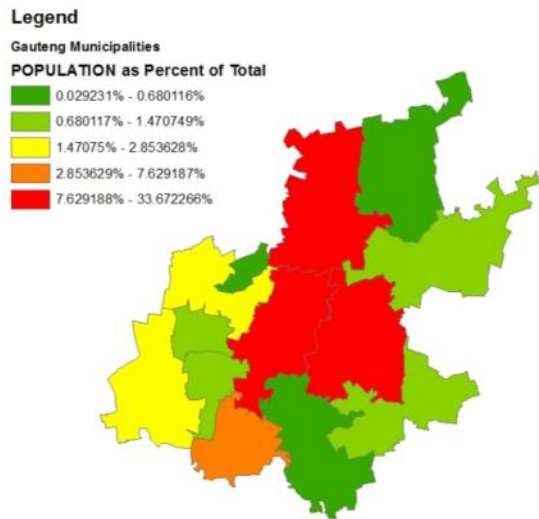
### **2.6.6. Dasymetric map**

A Dasymetric map is an alternative to the choropleth, but is used when the data is not uniformly distributed within enumeration units. Regions for this map type are then not predefined but rather chosen so that the distribution of the data is uniformed. The boundaries of the regions might be sharper than those of the Isarithmic map. The Dasymetric map is difficult to produce and thus not common.

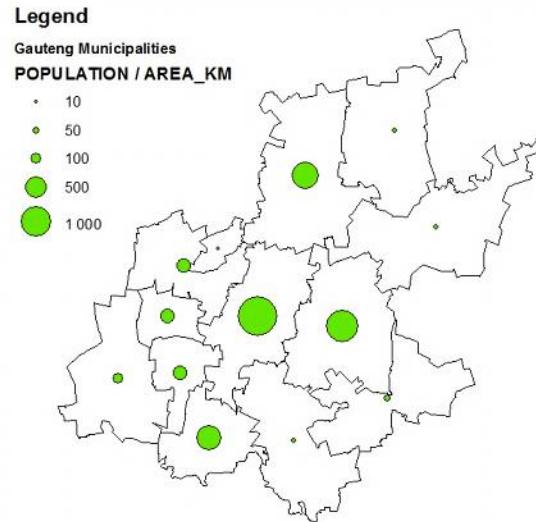
### **2.6.7. Flow map**

A flow map is the combination of a map and a flow chart, which depicts the movement of objects between geographic locations. The movement is typically represented with lines of varying widths (for

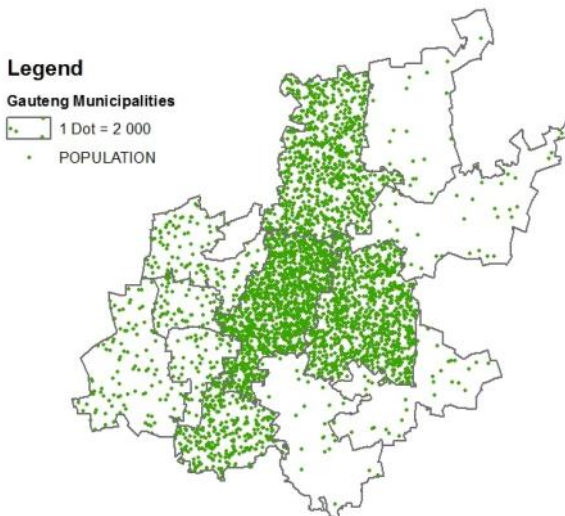
quantitative data), although other visual variables may be used. The flow map must not be confused with a route map, since the path between locations does not portray any routing information (Slocum et al., 2009).



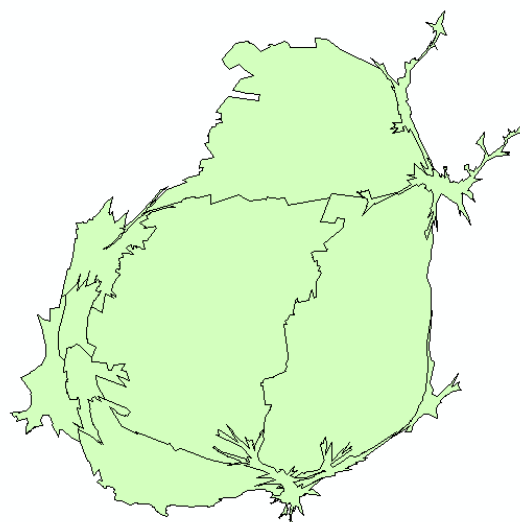
a) Choropleth map with five classes.



b) Proportional symbol map with five classes.

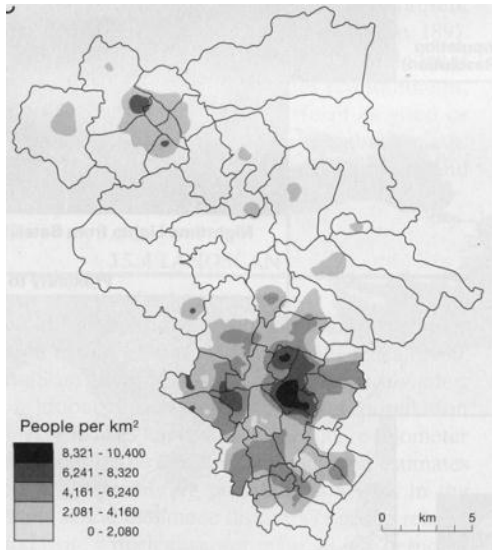


c) Dot density map

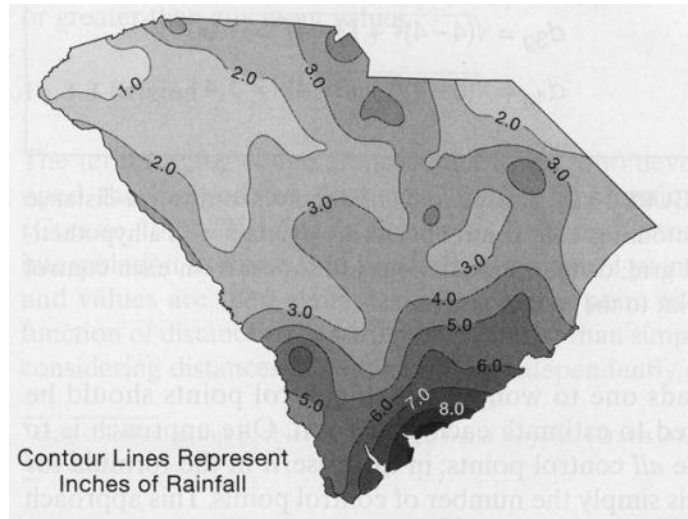


d) Cartogram

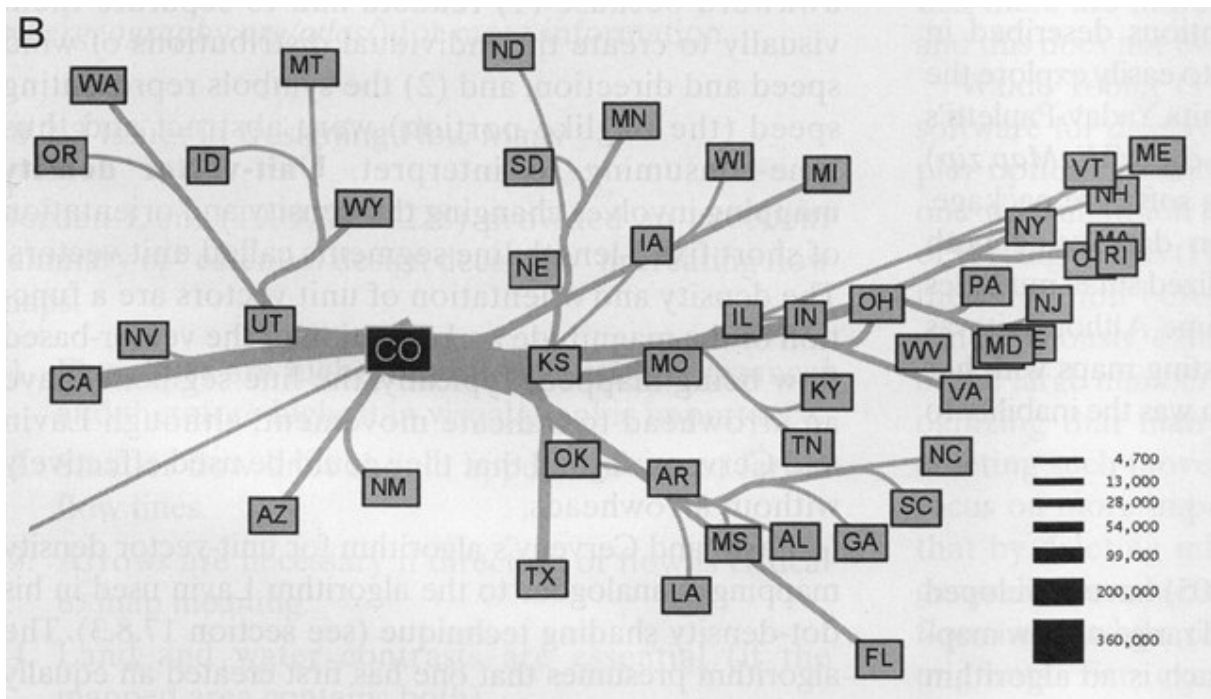
**Figure 6. The above maps are of the Gauteng province in South Africa depicting the population of each municipality**



a) Dasymetric map.



b) Isarithmic map.



c) Flow map.

Figure 7. Examples of the Isarithmic, Dasymetric map, and Flow map from Slocum et al. (2010)

## 2.7. Theoretical thematic cartography

In this dissertation only the choropleth and proportional symbol map are examined. In this section a brief description of the theoretical thematic cartography process of these types is provided.

### 2.7.1. Choropleth map

The first step of the choropleth process is to determine that the selected data is appropriate for the choropleth thematic map type. The ideal data is that which is uniformly distributed within each enumeration unit, changing only at the unit border of each enumeration. The next concern is the size

and shape of the enumeration units. This type of map works best and is most accurate when there is no significant variation in the size and shape of the units.

There are numerous different standardisation techniques available; these methods produce different views of the spatial pattern. It is the responsibility of the mapmaker to decide on the standardisation process and the selected process must be appropriate for the provided data and the message that is needed to convey to the audience.

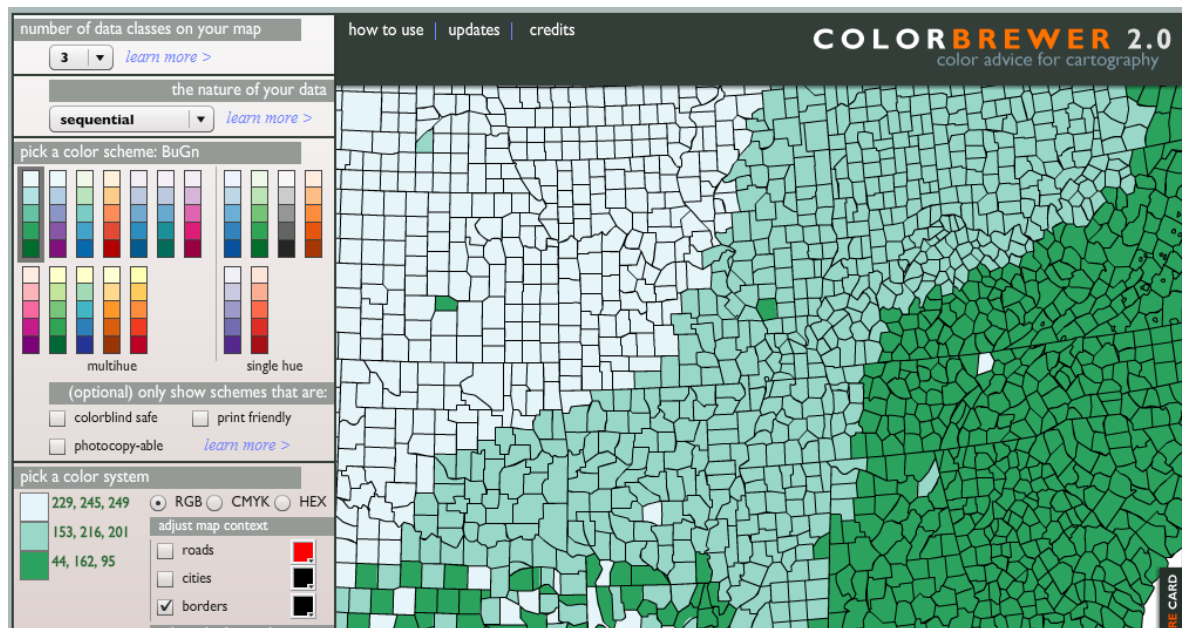
Next to be decided with respect to data classification is whether to class the data. The potential advantage of unclassified data is that it can in some instances portray data more accurately, and in contrast to this classified map, it is easier to process when there are only a limited number of categories. The deciding factor of classing your data is whether the map will be used for data exploration or communication to others. When communicating map data to others, a limited number of classes are the best option, and the same applies for bulk map production.

If the decision is to class the map, the mapmaker must now consider the range of classification and the actual method of classification. The following are some classification methods available: equal interval, quantiles, mean-standard deviation and optimal. With classification, the actual number of classes must also be specified. The optimal classification method is a good choice for novices since it assists you in the decision of selecting the appropriate number of classes.

A choropleth map relies highly on colour to convey its message to the audience. There are numerous factors that play a role in the selection of the colour scheme, but some of these factors will not be discussed. The kind of data to be mapped plays an important role in selecting a colour scheme. Brewer did significant research in this area and suggested that unipolar data sequential steps in the data should be represented with sequential steps in lightness of the selected colour; thus light colours convey low data values and darker colours for high values. As for bipolar data, two hues diverge from a common light hue or natural grey. There is some opposition for using spectral schemes; the reason for this is that yellow is a light colour and can cause some problems. An example of colours that can be used includes dark blue – bright yellow – dark red. There exists a website that was created by Brewer, called ColorBrewer, that provides a broad set of colour schemes for choropleth maps (refer to Figure 8).

It is also important to select colours that cannot be confused with each other; for example, red to pink is not optimal; however, red to blue is a good selection. Another consideration is users that are colour vision impaired, and the following website is useful to assess whether the selected colours will be optimal for colour vision impaired people.

The last consideration when producing a choropleth map is the legend design. This might seem like a minor point, but is actually considered one of the most important aspects of a map, since it provides the key to understanding the map and ensuring that the correct message is conveyed to the audience. There are two types of legends: a vertical legend, where the highest value is at the top, and a horizontal legend, where the values are written from left to right, lowest to highest.



**Figure 8. Home screen of the ColorBrewer website**

### 2.7.2. Proportional Symbol map

As with choropleth maps, it must first be determined whether the data is appropriate for this type of map. The following data types can be used: true point data, where measurements are taken at actual point locations, or conceptual data points, where data is collected over areas. With conceptual data points, it can be perceived that the data is located at the actual location of the point on the map.

Data standardisation is applicable as is described above for choropleth maps. There are two types of proportional symbols that can be used: geometric symbols, which are basic circle or square symbols that do not mirror the phenomenon being mapped, and pictographic symbols, which are pictures of clip art images that resemble the phenomenon. With pictographic symbols, overlapping may be difficult to interpret and deal with. To solve this problem, circles are frequently used. The reason for this is that they are visually stable, are preferred by users, and conserve map space. It is recommended that 3D images be avoided since they can be difficult to construct and establish size.

Scaling of the points must be done so that they are mathematically in direct proportion to the data. Classed or range graded maps, can be created by grouping the data and using a single symbol to represent the range of data values in the group. Unclassed proportional maps are, however, more popular. The choice of classed or unclassified can also be directly linked to the number of points that are to be mapped.

Again, the legend design is very important. There are two considerations that must be made with regard to the legend: how the symbols should be arranged, and which symbols should be included in the legend. Symbols can be arranged in two manners: nested, where smaller symbols are within larger symbols, or in a linear-legend, where symbols are placed adjacent to each other, horizontal or vertical. For classed maps all the symbols representing the classes can be added to the legend, and for unclassified maps at least the smallest and largest symbol must be in the legend.



As mentioned above, overlapping is of great concern in proportional maps. The first decision is how large symbols should be and how much overlay is allowed. When using small symbols, the problem of too much overlap can be avoided but can potentially decrease the spatial pattern. Conversely, the use of large symbols can produce a cluttered map and thus can be difficult to interpret. With regards to overlap, there is no rule; there is, however, some rule of thumb in that a map should not appear too empty or too full. Some overlap is important since it aids in the prominent of the spatial pattern. Overlap can also be handled using either transparent or opaque symbols.

## Chapter 3 Web services, OGC standards and Orchestration

### 3.1. Introduction

The chapter discuss general web services, the relevant Open Geospatial Consortium (OGC) web service, for example Web Map Service (WMS) and Web Feature Service (WFS), and other OGC standards such as Styled Layer Descriptor (SLD) and Common Query Language (CQL) that were used in the implementation of ThematicWS. Orchestration of web services and the technologies used to orchestrate web services, such as workflow modeller are described.

### 3.2. General web services

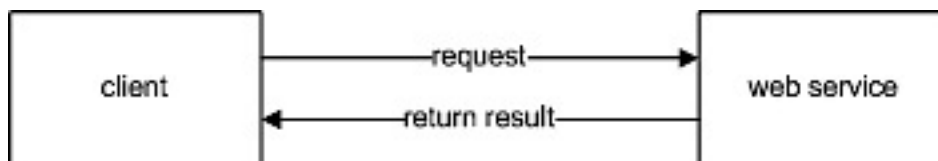
Web services are independent, self-describing, modular applications that can be published, located and dynamically invoked across the web (Weerawarana et al., 2005). They are entirely based on Extensible Markup Language (XML) technologies, and are composeable and discoverable based on their descriptions and terms and conditions available based on the web services metadata. A web service is thus any service that can be accessed over the Internet, uses XML to send and receive messages and is platform-independent (De Longueville, 2010). The user requires little to no knowledge about the implementation of the web service but has to be familiar with the interface of the service. The web service architecture allows a new service to be created from existing ones; this is a very valuable aspect of web services.

The definition of web services has evolved over the years. Early definitions were tied to Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL) and XML (Fu & Sun, 2010), since SOAP is no longer the only method to implement web service. Web services can be seen as a program that executes on a web server and exposes the programming interface to other applications on the web. Web services differ from web pages in that web pages are created for users to read and understand HyperText Markup Language (HTML) format whereas web services are created to expose web-based programmable components, and the result is usually JavaScript Object Notation (JSON) or XML. Web services have the following advantages over traditional computing techniques (Fu & Sun, 2010; Potts & Kopack, 2003):

1. The functionalities of the web service can be accessed without installing additional software on the local computer.
2. Platform independent, the web service is a kind of black box that only exposes its programming interface. Any operating system can be used to access it.
3. Loosely coupled, a web service can be consumed by any number of clients and coupled with different web services.
4. Legacy systems can easily be changed so that it can be wrapped in a web service.

- Updating the web service only has to be done on the server-side; all clients will have access to the new version.
- Lower operational costs (saves money on maintenance and hardware expenses, for example) and could produce new revenue opportunities.

For all of the advantages of web services there are also disadvantages (Potts & Kopack, 2003): availability of the web service (for example, problems with the internet service provider, time outs on the hosting server), and interface must be static (the interface cannot change too much, since this will affect the clients).



**Figure 9. Communication between web service and client**

Communication between the client and the web service is depicted in Figure 9. The format of the communication is used to categorise the web service, which can be either SAOP or Representational State Transfer (REST).

SOAP is a protocol specification for exchanging structured information in XML format (Fu & Sun, 2010; Weerawarana et al., 2005). SOAP became a World Wide Web Consortium (W3C) recommendation in 2003; however, this recommendation was removed in 2007 after SOAP was considered misleading. The requests and responses of these web services are SOAP encapsulated XML, and the HyperText Transfer Protocol (HTTP) POST are used to send requests to the web service. SOAP is often used in conjunction with WSDL.

REST is a software architecture that was introduced in 2000 by Ray Fielding as part of his doctoral dissertation (Fu & Sun, 2010; Weerawarana et al., 2005). REST was designed to take advantage of HTTP while reducing some of the complexity. REST refers to a collection of network architecture principles that outline how resources are defined and addressed. RESTful web services are services that transmit data over the HTTP without any additional messaging layer. In the most common RESTful architecture the client sends the request to the web service via a Uniform Resource Locator (URL), which contains all the parameters. In contrary to SOAP, REST does not define the server response; however, XML and JSON are still the most popular choices.

RESTful web services are the most common due to their simplicity and efficiency, which overshadows the rigorous discipline of SOAP. REST have a number of advantages over SOAP: it lowers the cost of developing web services for producers, reduces the cost of building GIS applications for users, and provides highly desirable architecture properties, such as scalability and performance, for managers.

A Web service can be discovered through a simple search mechanism and requires little or no knowledge about the web service to be able to consume it. Every web service has a set of operations

associated with the service that can be accessed through a standardised interface provided by the web service. As mentioned, a web services has a specific functionality and is deployed with appropriate quality of service at the end-point. The functional aspects of a web service are specified using WSDL. Policies and interfaces are used to define constrains and conditions of the web service and is attached to the WSDL. Information that is published about the service provides details of what the service is and does (thus the semantics).

WSDL is W3C standard that provides metadata about the web service, describing the web service in an XML format (Weerawarana et al., 2005). The XML document consists of three distinct parts: the envelope, header and body. The WSDL web service provides a document that describes the functional characteristics of the service, how to access them, the input parameters needed and the expected output. Another important aspect of web services is policies. Although WSDL and XML schema describe the functions of the web service, they do not, however, provide information about how the service delivers its interface or what is expected from the caller. Policies are very important when the aspect of service composition is discussed. SOAP and WSDL forms the basis of all W3C web services. They can be complex to implement, but there exist a number of tools that can automatically generate the application code to interface with WDSL or SOAP.

The Universal Description and Discovery Interface (UDDI) is the mechanism used to discover the web service and is a widely acknowledged specification of the web service registry. The registry can either be public, private, or semiprivate; this specifies who has access to the registry.

### **3.3. OGC web services**

#### **3.3.1. Standardisation organisations**

In the field of geographic information standards the International Standards Organisation (ISO) and the Open Geospatial Consortium (OGC) are the two major standardisation bodies.

OGC is an international industry consortium specialising in the development of geographic standards. The OGC standards are developed based on voluntary consensus and available free of charge to the public. The main goal of OGC is to come up with a set of specifications to be used as a guide to internet GIS design for different software vendors so that their designed systems can be communicated or interoperated with each other (Peng & Tsou, 2003).

ISO deals with the standardisation of all fields. The ISO is divided into a number of technical committees; the ISO/TC 211 is the technical committee that deals with geographic information standards. ISO standards are developed based on consensus and are currently the largest developer and publisher of standards. At present there are 64 countries involved in the ISO/TC 211, either in a participatory or observatory capacity. The main goals stated by the ISO/TC 211 are as follows: 'To increase the understanding and usage of geographical information; increase the availability, access, integration and sharing of geographic information; promote the efficient, effective and economic use of

digital geographic information and associated hardware and software systems; and contribute to a unified approach to addressing global ecological and humanitarian problems’.

ISO membership is free, but the standards are available at a small cost. ISO members are mainly from the public sector, including national standards bodies and organisations. OGC memberships are expensive, thus, most of the OGC members are from the private sector, including software vendors and GIS companies. Currently South Africa only has two members in the OGC: the CSIR and University of Pretoria.

ISO/TC 211 and OGC liaise closely and the cooperation is through the Joint Advisory Group (JAG). This committee ensures continuous cooperation in all phases of the development of standards in both organisations (Coetzee, 2008). At present the OGC does not have a voting seat in ISO/TC 211; however, the ISO/TC 211 chairman, Olaf Østensen, holds a voting seat at the OGC (Peng & Tsou, 2003).

### **3.3.2. OGC web services in general**

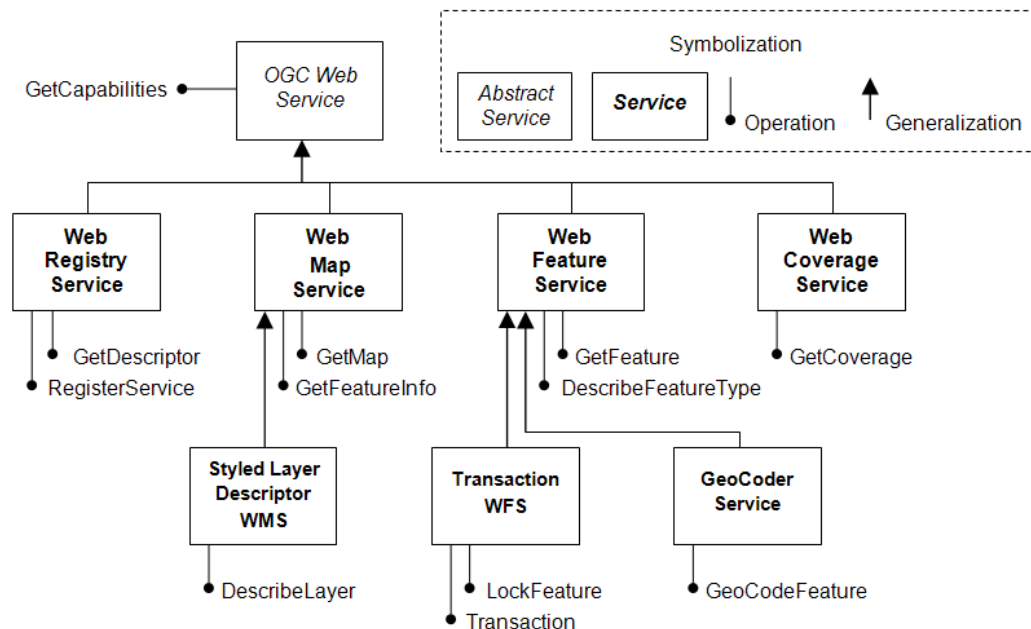
Open Geospatial Consortium (OGC) has recently emerged as a major force in the trend to openness in the geospatial committee. Geospatial services are generally implemented with OGC standards while non-spatial web services are not implemented based on these standards. Web mapping systems used to be proprietary systems that are isolated and do not allow interoperability. To address this problem, the OGC developed a non-propriety web mapping approach based on open-interfaces, encoding and schemas (OGC 2004). Refer to Figure 10 for the general architecture of the OGC web service standards.

The OGC Web Services are modular applications based on XML technology. These services are self-describing and can be published, located and invoked from anywhere on the Web or within any local network base. The OGC services are implemented and used mainly on the client side (Ioup et al., 2008). W3C and OGC web services are similar in that they rely on HTTP for transport and XML to express service descriptions, requests, and, in certain instances, data. For an OGC web service to be discoverable it must be added to the UDDI.

Web services and OGC services are independently developed standards and are not directly compatible (Ioup et al., 2008). If an OGC web service cannot be integrated with another web service, it is prohibited from the system; thus interoperability is important. The interoperability of web services relies heavily on uniformed standards. Interoperability, with respect to geoprocessing, refers to the ability of a digital system to exchange all kinds of spatial information over a network, as well as execute the necessary software capable of manipulating the information (OGC 2004).

The OGC develop and distribute specifications for the exchange and practical use of spatial data and the creation of geospatial web services. The standards provided by the OGC aids in the standardisation of web services that in turn assists with integration and interoperability of web services, thus interoperability relies heavily on interfaces and standard specifications. Unlike the W3C, the OGC standards are designed to handle a specific type of data, geographical data. W3C and OGC

web services are independently developed standards, but both depend on HTTP for transportation and XML for descriptions, requests and data in some instances (Ioup et al., 2008). While there are these similarities, W3C and OGC web services are not directly compatible. The most important feature of the OGC services is the uniformity of the service functionality. Each of the OGC standards have different request parameters, have different request types and return different data types.



**Figure 10. The OGC Web Services architecture (OGC 2002)**

Spatial web services can be categorised into three types: map services, data services, and analytical services. A map service is a type of service that allows the client to request a map image from a specific geographic extent. A data service allows the client to query, edit, and synchronise data over the web. A data service allows feature data to be edited, searching for spatial data using an index or catalogue; these are only a few functions of the data service. An analytical service performs a number of predefined GIS functions, for example network analysis and geocoding, via the web service.

### 3.3.3. Web Map Service (WMS)

Web Map Service (WMS) is a standard proposed by the OGC based on the HTTP GET or POST requests (Rita et al., 2010). When invoked, the service dynamically produces a map image from the geographic information and in accordance with the specifications provided by the user. The WMS is the most popular of the OGC network service, especially version 1.1.1; however, the WMS standard is currently at 1.3.0.

WMS service should be able to produce a map of a specific selection of data, answer basic queries about the content of the map, and provide clients with a list of other maps it can produce. The WMS service has the following four processing stages (Peng & Tsou, 2003):

1. The selection of the spatial data to be displayed.

2. The generation of display elements for the reference area.
3. The rendering of the display elements into a map.
4. The display of the map to the user.

WMS only allows the return of an image that cannot be edited or spatially analysed. The image is generated in raster format, like Portable Network Graphics (PNG), Graphics Interchange Format (GIF) or the Joint Photographic Experts Group (JPEG). These image formats support transparency allowing the user to overlay different datasets or themes, thus enabling the user to produce more complex maps.

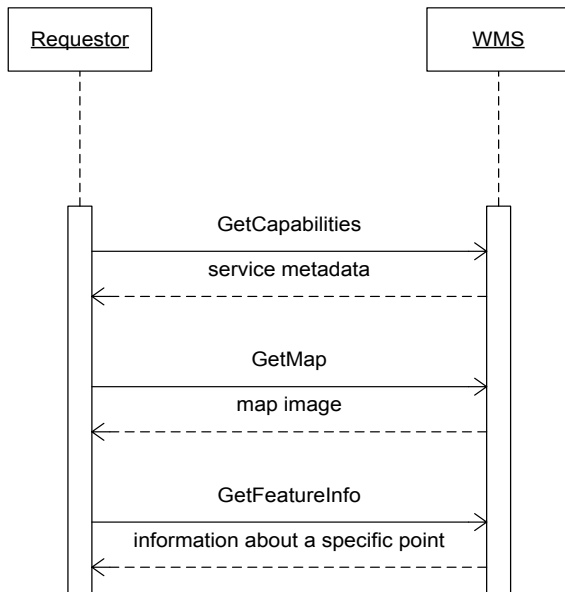
Within the WMS itself, there is no way of defining a unique style and a styling language is required, for example, Style Layer Descriptor (SLD). The WMS allows the user to specify a style using SLD. SLD is an extension of the WMS standard and is used to define the style or method in which the information is visually represented. The WMS can provide the name of the style, but it cannot tell the user what each portrayal will look like on the map.

According to the OGC, Web Services Commons (OWS) requires that all OGC web services support the GetCapabilities request. The WMS standard defines the following three operations (OGC 2006a):

1. GetCapabilities (Mandatory):  
The GetCapabilities returns an XML file containing the service metadata providing information about the service capabilities available to the client.
2. GetMap (Mandatory):  
The GetMap function returns the map image according to the specifications set in the parameters sent through (refer to Figure 12 as an example).
3. GetFeatureInfo (Optional):  
The GetFeatureInfo returns the information about a given point.

The GetCapabilities and GetMap operations are part of the basic WMS service. When the GetFeatureInfo operation is implemented, it is referred to as a *Queryable WMS*.

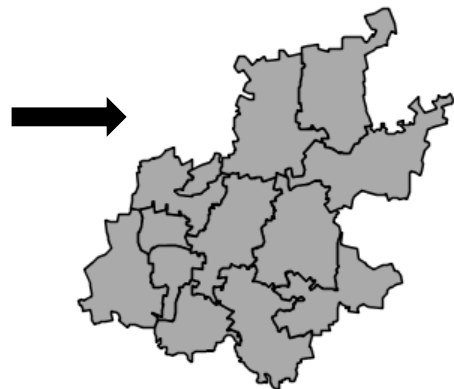
A service call includes the following parameters: list of layers, style, spatial reference system, width, height, format, background colour and transparency. A request can be submitted to the WMS service in the form of a URL. A WMS service can perform the following functions: produce a map, answer basic queries about the content of the map, or tell another program what maps it can produce and which of those can be queried further (OGC 2004).



**Figure 11. Basic sequence diagram depicting the different requests and responses of the WMS**

```

http://localhost:8080/geoserver/wms?service=WMS&version=1.1.0&request=GetMap&layers=postgis:gauteng_munic&styles=&bbox=27.2347,-26.2936,29.8543,-25.2923&width=501&height=512&srs=EPSG:4148&format=application/openlayers
  
```



**Figure 12. GetMap request example and the image it returns**

WMS has a couple of inconsistencies and challenges (Bernard et al., 2005):

1. Different Spatial Reference System (SRS)  
The problem with this is that interoperability can occur if multiple WMS does not share at least one SRS in common.
2. Multilingualism  
Different legends are produced by different WMSs.
3. Coherent use of scales  
WMS provides a general rule but no explicit reference on how to deal with different scales.
4. Consistent principles of cartographic styling  
At present WMS does not fully support SLD.

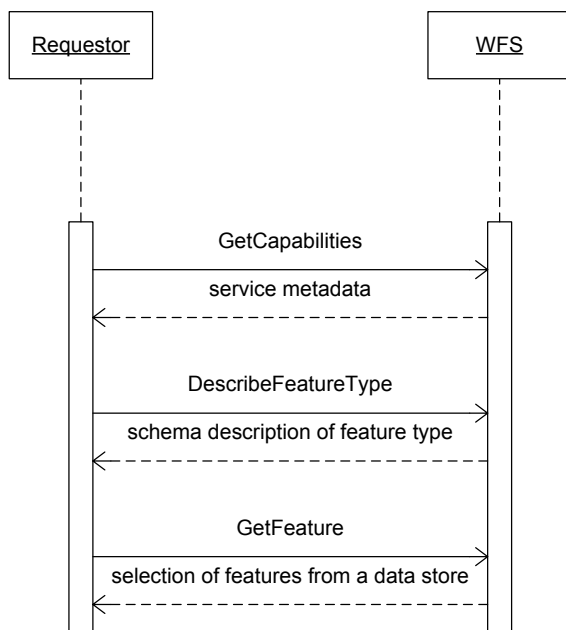


### 3.3.4. Web Feature Service (WFS)

Web Feature Service (WFS) is a web service standard proposed by the OGC. The WFS interface specifies an interface for the manipulating of geographical features. WFS brought a change in the way geographic information is created, modified and exchanged over the Internet independently of the underlying data store (OGC 2010b). Before the WFS standard, the File Transfer Protocol (FTP) was used. WFS offers direct access to geographic data at the feature and feature property level. The client can retrieve and modify the data requested through the use of WFS that is the standard that is used for data manipulation. Data manipulation includes the following:

1. Obtain or query features based on spatial and non-spatial constraints.
2. Create a new feature.
3. Delete a feature.
4. Update a feature.

As with other OGC web service standards, WFS uses HTTP request to access the web services' functionalities. The use of HTTP ensures that requests are platform independent. Geography Mark-up Language (GML) is used to pass data between the WFS service and the client. GML is an XML grammar defined by the OGC to express geographical features.



**Figure 13. Basic sequence diagram depicting the different requests and responses of the WFS**

The standard specifies eleven operations for the service, of which three are mandatory. WFS must implement the following operations according to the OGC specifications (OGC 2010b):

1. GetCapabilities  
The GetCapabilities operation generates service metadata for the WFS service. Encoding shall be in KVP-encoding and XML encoding is optional.
2. DescribeFeatureType

The DescribeFeatureType operation returns a schema description of feature type offered by a WFS instance.

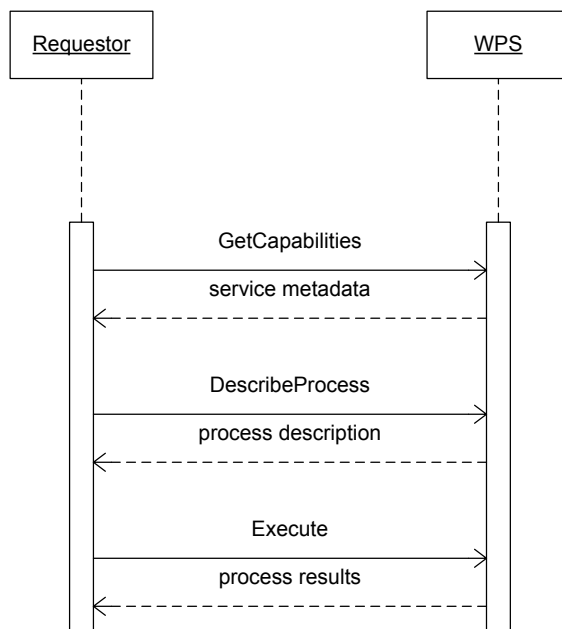
### 3. GetFeature

The GetFeature operation returns the selection of features from a data store.

The primary function of the WFS is the GetFeature operation that allows the client to access the set of features available from a server that all have client-desired property values. Data selected by one of the following feature property values or bounding box.

### 3.3.5. Web Process Service (WPS)

The web process service (WPS) specification was released in 2005 as a discussion paper (Foerster & Stoter, 2009). The purpose of the WPS is to describe a service that provides processing services that can be executed in a web environment. An unlimited variety of spatial processes can be described by a WPS. The WPS also provides a process description through its interface.



**Figure 14. Basic sequence diagram depicting the different requests and responses of the WPS**

WPS makes service chaining possible, but it is limited. Chaining is possible with the data retrieval and processing functionalities available in the WPS interface. WPS can be configured to offer any GIS functionality to a client across a network. The OGC WPS specification describes the following operations (OGC 2007b):

#### 1. GetCapabilities

The GetCapabilities returns an XML file containing the service metadata providing information about the service capabilities available to the client.

#### 2. DescribeProcess:

Allows the client to request and receive back detailed information about one or more processes that can be executed by the Execute operation and to include the input parameters and formats, as well as the outputs.

3. Execute:

This operation allows a client to run a specified process implemented by the WPS using provided input parameters and returning the outputs.

## 3.4. Orchestration of web services

### 3.4.1. Overview

Interoperability is the ability to communicate, execute programs, or transfer data amongst several functional units in such a way that the user requires little or no knowledge of the characteristics of the unit (ISO/IEC 2382-1). Interoperability is the freedom to use different components of an information system without compromising the overall success of the system (Yi et al., 1999). Standards are an important component required to achieve interoperability. With the use of standards an interface for the web service can be specified so that different users can use it. Web service standards essentially specify the format in which the HTTP requests and responses should be, so that the client can access the web service through the interface exposed. Interoperability in a geographic system refers to the ability of the system to freely exchange spatial data and cooperatively, over networks, execute software capable of manipulating the data (ISO 19119).

Web services can be composed in two different patterns (Sarang et al., 2007): orchestration and choreography. Web service orchestration is the automated arrangement of web services in a predefined pattern based on local decisions about their interactions with one another at the message and/or execution level (Sun et al., 2010; Suazo & Aguirre, 2005). Orchestration enables sophisticated forms of interaction with inherited intelligence or implicit automatic control. It uses a single process, referred to as a central controller, to control the underlying processes, execute the processes in the particular order in which they were specified and assign the necessary input parameters for the WPS processes. Choreography differs from orchestration in that there is no central controller, but rather each service involved is aware of the business process and with whom to interact and when to execute.

Web service 'chaining' and web service 'composition' are two examples of orchestration. These terms are sometimes used interchangeably - but are they really the same? According to ISO 19119 (2005), Yue et al., (2007) and Alameh (2003) service *chaining* is the process of combining monolithic services into a dependent series or pipeline to create a new web service. Web service *composition* on the other hand, according to Danapaquame and Bhavani (2010) and Zeng et al., (2003), refers to the combination of interoperable web services into a tree-like structure to create a new web service. Service chaining is thus a serial process (one web service is executed after the other) whereas service composition has a hierarchical nature (one service invokes one or more other services as part

of its execution). Web service chaining and composition both take advantage of existing web services to create new, more advanced web services. Web services, conforming to ISO and OGC standards, can be used to develop such advanced geospatial web services, either through chaining or through composition. Both chaining and composition of geospatial web services can provide the intelligence required in an SII's intelligent geoportal.

There exist two levels of web service interoperability: syntactical and semantical interoperability (ISO 19119, Yue et al., 2007). Syntactical interoperability is a technical connection which provides a connection that enables the transference of data between web services. This type of interoperability can be achieved through the use of common web standards, like WSDL and SOAP. Semantical interoperability assures that the contents of data and services are correctly understood when data and services are connected. Semantical interoperability can be accomplished by means of ontologies. Ontology is a formal specification of a common vocabulary and defines the meaning and relationship between them.

Services that are semantically interoperable have a common understanding of the information that is processed and transferred between the different services. With semantic services a number of functions, such as service discovery, composition, and execution can be semi-automated or fully automated making it possible to perform complex functionalities (Cardoso and Sheth 2005). Semantically interoperable service are required to solve more complex problems where, for example, a relevant service needs to be discovered at run-time. The addition of semantics to services will provide the ability to change workflows in runtime according to the outcome of previous services.

Web service composition is currently not standardised; however, service chaining is covered by ISO 19119:2005. Three types of user interaction are defined in service chaining (ISO 19119; Foerster et al., 2010):

1. Transparent

Also known as user defined chaining. This type of interaction requires full (human) user interaction and prior knowledge of the services in the chain. It is the simplest way of web service interaction.

2. Translucent

Also known as workflow-managed chaining. The user invokes the chain and is aware of the interaction, but cannot interfere or reorder the services.

3. Opaque

Also known as aggregate service. The user is not aware of the service chain; he/she sees it only as a single service.

Business Process Model and Notation (BPMN) is a notation language first introduced as the Business Process Modeling Language (BPML) standard. BPMN is a graphical notation/language, which depicts the steps in a business process or workflow (OMG 2011). The notation was designed to coordinate

the sequence of services in a process and the flow of messages between the different participants. BPMN is similar to BPEL in that it is an XML based language, for process flow control and activities.

### **3.4.2. Orchestration languages**

A number of orchestration languages exist for the orchestration of web services. In this section a brief overview of the following languages are provided: BPEL, XPD, WSMO, and WSCI.

#### **3.4.2.1. Business process execution language (BPEL)**

Business process execution language (BPEL) is an executable language standard that composes services by choreographing service interactions (Weerawarana et al., 2005), developed by the organisation for the advancement of structured information standards (OASIS). It is an XML-based block-structured language used to describe the control logic for web service orchestration in a business process (OASIS 2003). Complex web services can be created from smaller processes and services (Sarang et al., 2007). BPEL makes use of web service standards as a basis. The final process created with BPEL is exposed as a web service through WSDL, which describes the public entry and exit point for a process. BPEL interacts with external web services through the WSDL interface and makes use of the WSDL data types to describe information flow.

BPEL allows the creation of a centralised process, which invokes all the containing processes in sequential order, thus controlling the entire workflow. Each step in the process is called an *activity* and contains some functionality, for example, *invoke*, *reply* or *terminate*. These functionalities together with BPEL structural activities, for example, *switch* and *while* are used together to create complex business processes. BPEL allows the creation of complex processes based on a syntactical connection between web services, but does not yet support web services connected through semantics. BPEL is the *de facto* standard for service orchestration.

In the related work section (refer to Section 5.4.2), the use and limitations of using BPEL for the orchestration of OGC web services are described.

#### **3.4.2.2. XML Process Definition Language (XPDL)**

XML Process Definition Language (XPDL) is an XML-based language developed by the Workflow Management Coalition (WfMC) used to define business processes. XPDL can be seen as a graph-structured language, which does not allow nested process definitions. XPDL packages correspond to process diagrams in a BPMN and consist of a set of process definitions. XPDL was used successfully in the implementation of the workflow of RESTful services for the OGC OWS-6 testbed (OGC 2009).

#### **3.4.2.3. Web Service Modeling Ontology (WSMO)**

Workflow modelling has become a popular topic within the semantic web; Web Service Modeling Ontology (WSMO) is a solution to the semantic discovery and composition of web services. WSMO is a generic and extendable language that offers complete conceptualisation for the description, composition and execution of semantic web services (Gone & Schade, 2007). WSMO consists of four elements: WSMO ontologies, WSMO web services, WSMO goals, and WSMO mediators.

Orchestration of OGC web services with WSMO is currently limited, but has been identified as a potential future research area.

#### **3.4.2.4. Web Service Choreography Interface (WSCI)**

Web Service Choreography Interface (WSCI) facilitates the composition and orchestration of the execution of existing services to create more powerful ones. W3C recommended the WSCI standard in 2002 (W3C 2002). WSCI is the first technology of its kind submitted to a standards body for describing the interactive behavioural aspects of a web service. WSCI or Web Service Flow Language (WSFL) is an XML based business process modelling language that describes the collaboration protocols of cooperating Web Services. A WSDL operation is the entry point into a service, thus each unit of work in the WSCI is a WSDL operation. This allows the user to chain a number of services together, but there is no central controller.

### **3.5. Styled Layer Descriptor (SLD)**

#### **3.5.1. Overview of SLD**

Styled Layer Descriptor (SLD) is an XML schema that defines the structure of a layer style. It is an OGC standard to define styles that can be used for publishing raster and vector data on the Internet. The main purpose of the SLD is to allow users to define their own styles. SLD is what makes the map more colourful and user-friendly and it is responsible for telling the server how to render the map. SLD is robust enough to fulfil a wide range of cartographic needs and uses the HTTP as a transport method.

Thematic maps can be produced with the use of SLD; these maps are, however, static and the generation of more complex maps can be very demanding and time-consuming (Woźniak et al., 2011). However, there are two major problems when producing thematic maps with SLD style sheets: the creation of such a large style sheet is very extensive, and the additional burden on the WMS server while generating a graphic representation.

The second problem can be solved with the use of a WMS-C. The WMS-C is an extension of the WMS standard that cache images the server generates. The WMS-C thus allows for the early preparation of visualisation in the cache and then prepares to share this image. Unfortunately, this is only possible if the presentation of static data in specific spatial scales and ranges. An SLD WMS adds the following additional operations to that of the basic WMS (OGC 2007d):

1. DescribeLayer

When describing a user-defined style some information about the feature being described is necessary. This function provides an XML description of the map layer.

2. GetLegendGraphic

A legend is a very important part of a map. It provides information about how features are represented on the map. The function is used to acquire legend symbols and the response is in the form of a picture.

### 3. GetStyles

The function is used to retrieve user-defined styles from a WMS.

### 4. PutStyles

The function is used to store user-defined styles into a WMS so that it can be retrieved at a later stage.

An SLD style sheet is a sequence of layered elements. The first element is the `StyledLayerDescriptor` which has two sub elements: `NamedLayer` and `UserLayer`. The `NamedLayer` element is used to refer to a style that can be accessed in a WMS, whereas the `UserLayer` allows a user-defined layer to be built from WFS and WCS data. The `NamedLayer` is focused on here, since the WMS service is used to produce the map.

The `UserStyle` is used to create the user-defined style, which contains a title and a short abstract to describe the style. The `FeatureTypeStyle` contains the styling definition of one feature that is described using rule elements. Each rule element has a name and two sub-elements: filter and symbolizer. There are a number of symbolizers available: `LineSymbolizer`, `PolygonSymbolizer`, `PointSymbolizer`, `TextSymbolizer`, and `RasterSymbolizer`. Within these symbolizers a number of properties can be set, for example, the geometry, fill, and stroke for the `PolygonSymbolizer`. These properties can be set using `CssParameter`, which refers to a Scalable Vector Graphics (SVG) or Cascading Style Sheets (CSS) graphic formatting parameter. This is just a short description of some of the elements in the SLD style sheets (refer to the figures in the Section 3.5.2 for more information).

#### 3.5.2. Creating thematic maps with SLD

A thematic map can be produced using thematic styling with the use of standard SLD. The class boundaries, mean, standard deviation and other required statistical information will need to be calculated manually by the user before the SLD style sheet can be created. The thematic maps produced with SLD are usually static, in that their styles and classes cannot be changed at run-time; however, it is possible to generate these style sheets dynamically, which is very demanding and time-consuming. Every class of the thematic map will be represented with a rule in the SLD, and for unclassed maps every unique value will require a rule. Each of these rules has to be either prepared in a separate tool and exported to an SLD, or written manually by a user.

Figure 15 shows the style sheet required to produce a choropleth thematic map with three classes. The produced map is shown in Figure 16.

```
1 <?xml version="1.0" encoding="ISO-8859-1"?>
2 <StyledLayerDescriptor version="1.0.0" xmlns=http://www.opengis.net/sld
3   xmlns:ogc="http://www.opengis.net/ogc"
4   xmlns:xlink="http://www.w3.org/1999/xlink"
5   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6   xsi:schemaLocation="http://www.opengis.net/sld
```

```

7 http://schemas.opengis.net/sld/1.0.0/StyledLayerDescriptor.xsd"
8 <NamedLayer>
9   <Name>Thematic Style</Name>
10  <UserStyle>
11    <Title>Thematic SLD Style</Title>
12    <Abstract>Basic SLD style to illustrate how thematic map is produce
13      with SLD</Abstract>
14    <FeatureTypeStyle>
15      <Rule>
16        <Name>SmallPop</Name> <Title>Less Than 200,000</Title>
17        <ogc:Filter>
18          <ogc:PropertyIsLessThan>
19            <ogc:PropertyName>population</ogc:PropertyName>
20            <ogc:Literal>200000</ogc:Literal>
21          </ogc:PropertyIsLessThan>
22        </ogc:Filter>
23        <PolygonSymbolizer>
24          <Fill>
25            <CssParameter name="fill">#00FF00</CssParameter>
26          </Fill>
27          <Stroke>
28            <CssParameter name="stroke">#000000</CssParameter>
29            <CssParameter name="stroke-width">1</CssParameter>
30          </Stroke>
31        </PolygonSymbolizer>
32      </Rule>
33      <Rule>
34        <Name>MediumPop</Name> <Title>200,000 to 2,000,000</Title>
35        <ogc:Filter>
36          <ogc:And>
37            <ogc:PropertyIsGreaterThanOrEqualTo>
38              <ogc:PropertyName>population</ogc:PropertyName>
39              <ogc:Literal>200000</ogc:Literal>
40            </ogc:PropertyIsGreaterThanOrEqualTo>
41            <ogc:PropertyIsLessThan>
42              <ogc:PropertyName>population</ogc:PropertyName>
43              <ogc:Literal>2000000</ogc:Literal>
44            </ogc:PropertyIsLessThan>
45          </ogc:And>
46        </ogc:Filter>
47        <PolygonSymbolizer>
48          <Fill>
49            <CssParameter name="fill">#FFFF00</CssParameter>
50          </Fill>
51          <Stroke>
52            <CssParameter name="stroke">#000000</CssParameter>
53            <CssParameter name="stroke-width">1</CssParameter>
54          </Stroke>
55        </PolygonSymbolizer>
56      </Rule>
57      <Rule>
58        <Name>LargePop</Name> <Title>Greater Than 2,000,000</Title>
59        <ogc:Filter>
60          <ogc:PropertyIsGreaterThan>
61            <ogc:PropertyName>population</ogc:PropertyName>
62            <ogc:Literal>2000000</ogc:Literal>
63          </ogc:PropertyIsGreaterThan>
64        </ogc:Filter>
65        <PolygonSymbolizer>
66          <Fill>
67            <CssParameter name="fill">#FF0000</CssParameter>
68          </Fill>
69          <Stroke>
70            <CssParameter name="stroke">#000000</CssParameter>
71            <CssParameter name="stroke-width">1</CssParameter>
72          </Stroke>
73        </PolygonSymbolizer>
74      </Rule>
75    </FeatureTypeStyle>
76  </UserStyle>
77 </NamedLayer>

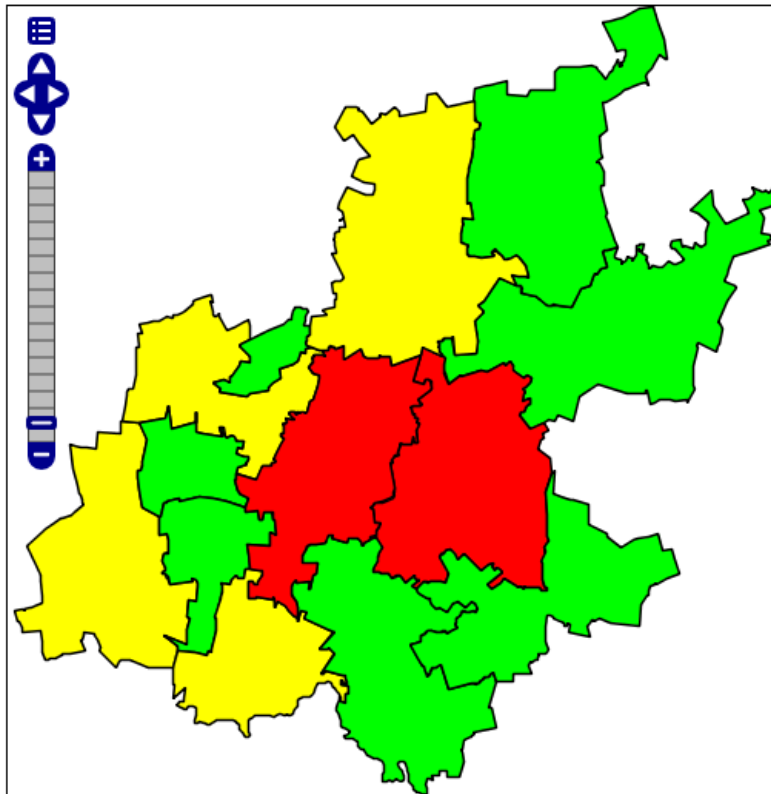
```



```

78     </FeatureTypeStyle>
79     </UserStyle>
80   </NamedLayer>
81 </StyledLayerDescriptor>
  
```

**Figure 15. SLD code to produce a choropleth map with three classes, adapted from the SLD Cook book (Pumphrey, 2010)**



**Figure 16. The resulting thematic map of the Gauteng municipalities based on population**

Figure 17 shows the style sheet required to produce a proportional symbol map with three classes, the produced map is shown in Figure 18. In this style sheet there is a polygon symbolizer and a point symbolizer; without the polygon symbolizer, only points will be shown on the map without polygon boundaries.

```

1   <?xml version="1.0" encoding="ISO-8859-1"?>
2   <StyledLayerDescriptor version="1.0.0"
3     xsi:schemaLocation="http://www.opengis.net/sld StyledLayerDescriptor.xsd"
4     xmlns="http://www.opengis.net/sld"
5     xmlns:ogc="http://www.opengis.net/ogc"
6     xmlns:xlink="http://www.w3.org/1999/xlink"
7     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
8     <NamedLayer>
9       <Name>Attribute-based point</Name>
10      <UserStyle>
11        <Title>GeoServer SLD Cook Book: Attribute-based point</Title>
12        <FeatureTypeStyle>
13          <Rule>
14            <Name>SmallPop</Name>
15            <Title>1 to 50000</Title>
16            <ogc:Filter>
  
```

```

17         <ogc:PropertyIsLessThan>
18             <ogc:PropertyName>POPULATION</ogc:PropertyName>
19             <ogc:Literal>200000</ogc:Literal>
20         </ogc:PropertyIsLessThan>
21     </ogc:Filter>
22     <PointSymbolizer>
23         <Graphic>
24             <Mark>
25                 <WellKnownName>circle</WellKnownName>
26                 <Fill>
27                     <CssParameter name="fill">#0033CC</CssParameter>
28                 </Fill>
29             </Mark>
30             <Size>15</Size>
31         </Graphic>
32     </PointSymbolizer>
33     <PolygonSymbolizer>
34         <Stroke>
35             <CssParameter name="stroke">#000000</CssParameter>
36             <CssParameter name="stroke-width">1</CssParameter>
37         </Stroke>
38     </PolygonSymbolizer>
39 </Rule>
40 <Rule>
41     <Name>MediumPop</Name>
42     <Title>50000 to 100000</Title>
43     <ogc:Filter>
44         <ogc:And>
45             <ogc:PropertyIsGreaterThanOrEqualTo>
46                 <ogc:PropertyName>POPULATION</ogc:PropertyName>
47                 <ogc:Literal>200000</ogc:Literal>
48             </ogc:PropertyIsGreaterThanOrEqualTo>
49             <ogc:PropertyIsLessThan>
50                 <ogc:PropertyName>POPULATION</ogc:PropertyName>
51                 <ogc:Literal>2000000</ogc:Literal>
52             </ogc:PropertyIsLessThan>
53         </ogc:And>
54     </ogc:Filter>
55     <PointSymbolizer>
56         <Graphic>
57             <Mark>
58                 <WellKnownName>circle</WellKnownName>
59                 <Fill>
60                     <CssParameter name="fill">#0033CC</CssParameter>
61                 </Fill>
62             </Mark>
63             <Size>19</Size>
64         </Graphic>
65     </PointSymbolizer>
66     <PolygonSymbolizer>
67         <Stroke>
68             <CssParameter name="stroke">#000000</CssParameter>
69             <CssParameter name="stroke-width">1</CssParameter>
70         </Stroke>
71     </PolygonSymbolizer>
72 </Rule>
73 <Rule>
74     <Name>LargePop</Name>
75     <Title>Greater than 100000</Title>
76     <ogc:Filter>
77         <ogc:PropertyIsGreaterThanOrEqualTo>
78             <ogc:PropertyName>POPULATION</ogc:PropertyName>
79             <ogc:Literal>200000</ogc:Literal>
80         </ogc:PropertyIsGreaterThanOrEqualTo>
81     </ogc:Filter>
82     <PointSymbolizer>
83         <Graphic>
84             <Mark>

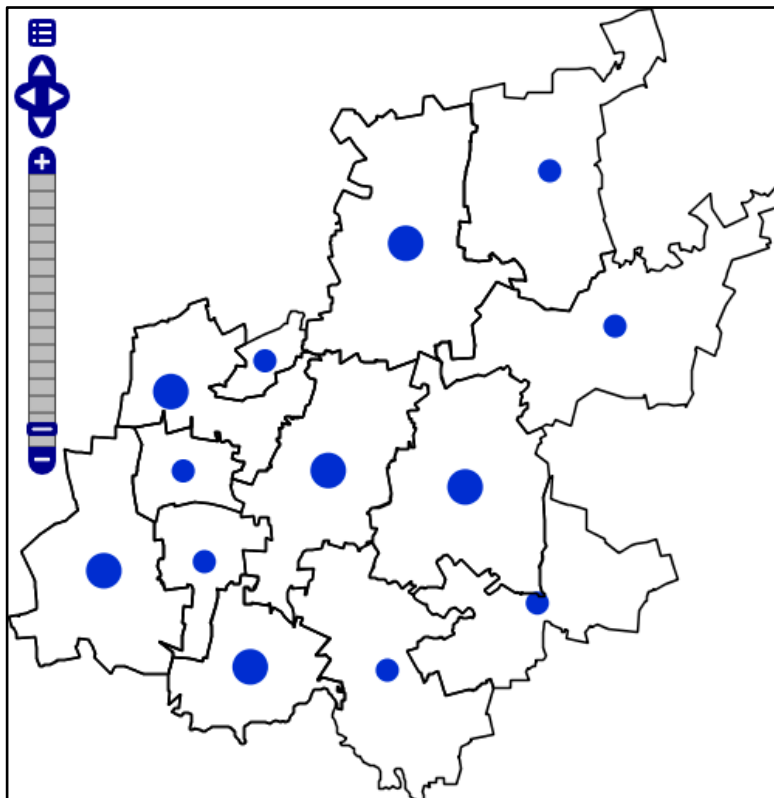
```

```

88         <WellKnownName>circle</WellKnownName>
89         <Fill>
90             <CssParameter name="fill">#0033CC</CssParameter>
91         </Fill>
92         </Mark>
93         <Size>23</Size>
94     </Graphic>
95 </PointSymbolizer>
96 <PolygonSymbolizer>
97     <Stroke>
98         <CssParameter name="stroke">#000000</CssParameter>
99         <CssParameter name="stroke-width">1</CssParameter>
100    </Stroke>
101 </PolygonSymbolizer>
102 </Rule>
103 </FeatureTypeStyle>
104 </UserStyle>
105 </NamedLayer>
106 </StyledLayerDescriptor>

```

**Figure 17. SLD code to produce a proportional symbol map with three classes, adapted from the SLD Cook book (Pumphrey, 2010)**



**Figure 18. The resulting thematic map of the Gauteng municipalities based on population**

### 3.5.3. Styled Layer Descriptor (SLD) extensions in GeoServer

GeoServer is an open source Java-based server that allows geospatial data to be displayed and edited (OpenGeo, 2009). GeoServer is built on the GeoTools API and implements OGC standards. According to the SourceForge (2011) statistics, GeoServer can boast a minimum of 10,000 downloads per month over the last year, showing that GeoServer is a popular and widely used server

for geospatial data. GeoServer can display data with a WMS over the Web in multiple output formats and the SLD can be used to create custom styles. The OpenLayers library, which is the *de facto* application programming interface (API) standard for open source web mapping, provides the functionality to display map data in web browsers making map generation quick and easy. OpenLayers is integrated into GeoServer. The implementation of WFS in GeoServer allows the sharing and editing of the data that is used to produce the maps. GeoServer allows seamless integration with multiple popular mapping applications such as Google Maps, Google Earth, Yahoo Maps and Microsoft Virtual Earth.

Rita et al., (2010) constructed Table 2 depicting the different levels of difficulties in producing a thematic map with GeoServer, SLD and SE.

**Table 2. Comparison between the different types of thematic maps (Rita et al., 2010)**

	<b>Difficulty of producing thematic maps using SLD and SE standards</b>
Choropleth	Moderate
Diagrams	Hard
Proportional Symbols	Easy
Overlaid Symbols	Easy
Juxtaposed Symbols	Impossible
Cartogram Maps	Impossible

Combining WMS, SLD and Symbol Encoding (SE) provides a viable open framework for basic topographic representations, but has some limitations for producing high quality cartographic limitation (Iosifescu-Enescu et al., 2010). For example, SLD does not support user defined point, multilayered symbols, or pattern which are often used by cartographers. The extensions in GeoServer provide support for some of these limitations.

### **3.5.3.1. Dynamic symbolizers**

The dynamic symbolizer was implemented to overcome some of the limitations of SLD. This extension was first implemented in GeoServer 1.7. The dynamic symbolizer is an extension of the PointSymbolizer described in OGCs SLD (2007) standard and provides the user with the following three additional functionalities (Jordan, 2010):

1. Creates external references that contain feature attributes as variables.
2. Uses decorative true type fonts as markers in the map.
3. Programs one's own dynamic symbolizers that extend existing ones and have full access to an individual feature's attributes at run-time.

The Mark element of the PointSymbolizer in the standard SLD is used to specify the graphic element on the thematic map. A well-known name such as 'square' or 'circle' can be used in the Mark element. The ExternalGraphic element of the PointSymbolizer in the standard SLD specifies a URL to a GIF or PNG image to be used on the thematic map (Woźniak et al., 2011). In standard SLD, both Mark and

ExternalGraphic are static elements, which imply that they cannot vary according to the unique features in the data set (i.e. all features on the thematic map are represented with the same Mark or ExternalGraphic). A user can now embed attribute names into these elements for value expansion at run-time, i.e. the value of the attribute at run-time determines the value of the Mark or ExternalGraphic element (thus the thematic style).

Another feature of the dynamic symbolizer is that valid CQL expression can be embedded in the WellKnownName sub-element of the Mark element and the OnlineResource/@xlink:href sub-element of the ExternalGraphic element. Style sheets can be created more compactly with this functionality, in cases where the symbol name could be derived from the attribute value (GeoServer, 2011). The dynamic symbolizer can effectively be used for proportional symbol maps since it uses the PointSymbolizer required for this type of thematic map. A choropleth map, however, requires the use of the PolygonSymbolizer, for which the dynamic symbolizer is not implemented.

Figure 19 is an example of the dynamic symbolizer used within the OnlineResource element that contains a CQL expression, which is evaluated at run-time. This method is optimal for unclassed symbol maps.

```

1   <FeatureTypeStyle>
2     <Rule>
3       <Title>Dynamic Symbolizer Example</Title>
4       <PointSymbolizer>
5         <Graphic>
6           <ExternalGraphic>
7             <OnlineResource xlink:type="simple" xlink:href=
8               "http://mysite/geoserver_images/${municipal_type}.jpg"/>
9             <Format>image/gif</Format>
10          </ExternalGraphic>
11        </Graphic>
12      </PointSymbolizer>
13      <PolygonSymbolizer>
14        <Stroke>
15          <CssParameter name="stroke">#000000</CssParameter>
16          <CssParameter name="stroke-width">0.5</CssParameter>
17        </Stroke>
18      </PolygonSymbolizer>
19    </Rule>
20  </FeatureTypeStyle>

```

**Figure 19. Example of a style sheet implementation of the dynamic symbolizer**

### 3.5.3.2. Parameter substitution in SLD

This extension to standard SLD provides the ability to pass parameter values in the WMS request to the SLD style sheet. This allows the user to dynamically change colours, fonts, sizes and filter thresholds in the SLD style sheet (GeoServer, 2011). A list of env parameters can be added to the GetMap request of a WMS service:

```
...&env=paramName:value;otherParam=otherValue&...
```

The env function can be specified in any element of an SLD rule where an OGC expression is used such as in the size, offsets, filter and CssParameter elements. An example of such a function in a

style sheet for a choropleth map is provided in Figure 20, which uses the env function in the filter and CssParameter. In the filter, the env parameter adjusts the lowerlimit (Figure 20, line 7-10) and lowercolour (Figure 20, line 16-19) parameter, which changes the class limit in the rule and the colour assigned to the area. Default values for each parameter are specified (200,000 and 00FF00). The default value is used if the parameter is not passed with the GetMap request.

```

1  <Rule>
2    <Name>SmallPop</Name>
3    <Title>Less Than 200,000</Title>
4    <ogc:Filter>
5      <ogc:PropertyIsLessThan>
6        <ogc:PropertyName>population</ogc:PropertyName>
7        <ogc:Function name="env">
8          <ogc:Literal>lowerlimit</ogc:Literal>
9          <ogc:Literal>200000</ogc:Literal>
10       </ogc:Function>
11     </ogc:PropertyIsLessThan>
12   </ogc:Filter>
13   <PolygonSymbolizer>
14     <Fill>
15       <CssParameter name="fill">
16         <ogc:Function name="env">
17           <ogc:Literal>lowercolour</ogc:Literal>
18           <ogc:Literal>00FF00</ogc:Literal>
19         </ogc:Function>
20       </CssParameter>
21     </Fill>
22     <Stroke>
23       <CssParameter name="stroke">#000000</CssParameter>
24       <CssParameter name="stroke-width">1</CssParameter>
25     </Stroke>
26   </PolygonSymbolizer>
27 </Rule>

```

**Figure 20. Example of style sheet implementing parameter substitution**

### 3.5.3.3. Chart extension

The chart extension combines the functionalities of dynamic symbolizers and Google's Charts API (described in Section 6.2.3.6) and makes the development of diagram maps possible (GeoServer 2011). Refer to the example in Figure 21.

```

1  <Rule>
2    <PointSymbolizer>
3      <Graphic>
4        <ExternalGraphic>
5          <OnlineResource
6            xlink:href="http://chart?cht=p&chd=
7              t:${100*employed/population},
8              ${100 * unemployed / population}&chf=bg,s,FFFFFF00" />
9          <Format>application/chart</Format>
10         </ExternalGraphic>
11       <Size>
12         <ogc:Add>
13           <ogc:Literal>20</ogc:Literal>
14           <ogc:Mul>
15             <ogc:Div>
16               <ogc:PropertyName>population</ogc:PropertyName>
17               <ogc:Literal>20000000.0</ogc:Literal>
18             </ogc:Div>
19           <ogc:Literal>50</ogc:Literal>

```

```

20         </ogc:Mul>
21         </ogc:Add>
22         </Size>
23     </Graphic>
24 </PointSymbolizer>
25 </Rule>
  
```

**Figure 21. An example of a diagram map produced with the GeoServer charts extension**

### 3.5.4. Styled Layer Descriptor (SLD) current work in OGC

Secondly, the GeoServer extensions to SLD automate some of this programming, but relying on them limits interoperability, because the dynamic symbolizer, for example, is not a standard. Thus, a solution implementing the SLD extensions cannot readily be ported to another OGC compliant WMS implementation, i.e. independent intelligent orchestration of web services is not possible. Hopefully, the on-going work in the OGC standards working group on 'SLD and SE 1.2' will solve some of these problems. At present, the standards working group 'SLD and SE 1.2' are busy with integrating a DiagramSymbolizer (similar to the PointSymbolizer) into SLD. The DiagramSymbolizer allows the user to visualise multiple data values using diagrams. The work on the new version of SLD and SE was prompted by numerous change requests to extend SLD and SE for improved thematic mapping, some as early as 2007. A recent change request posted in the 2011 re-voting process proposes the addition of classification methods to the SLD standard. However, implementing these methods requires the DescribeLayer operation in WMS to contain Count, Maximum, Minimum, Sum, Mean and StandardDeviation properties for every ogc:PropertyName element. Therefore, the revised WMS (currently being prepared) has to precede that of the revised SLD. The GeoServer development team submitted a change request in 2011 to include all the GeoServer extensions and improvements to SLD. Work on these requests is still in progress. In addition, the extensions to SLD in GeoServer are not comprehensive, for example, CQL expressions in the LineSymbolizer and PolygonSymbolizer are not supported, but OGC is not yet looking at these.

### 3.6. Symbol Encoding (SE) and Filter Encoding (FE)

Symbol Encoding (SE) was originally part of SLD, but was separated from SLD in 2007 (OGC 2006b). SE is an XML scheme capable of describing the method in which vector and raster data should be rendered. SE allows the user to define rules and methods in order to display objects. The elements also contain certain conditions for filtering the objects in accordance with the specifications of the filter encoding. Symbolizer elements describe not only the shape of what should appear but also the properties of graphic, such as colour and opacity of visualised objects.

Filter Encoding (FE) was part of the WFS originally, but due to the potential use of FE with other services it was separated. WFS, WCS and Web registries mainly use the FE (OGC 2010a). FE allows the selection of map objects based on conditions imposed on their attributes. The order in which filters

are created is very important. Subsequent filters can be imposed on each other. FE is an XML encoding of the OGC Common Query Language (CQL).

### **3.7. Common Query Language (CQL)**

Common Query Language (CQL) is a formal language developed by the United States Library of Congress for information retrieval (The Library of Congress 2003). CQL tries to combine the simplicity and intuitiveness of Google searching with the expressive power of XQuery. It can support very simple queries and arbitrarily complex expressions as necessary. The design objectives of CQL queries is to be easily human readable and writeable. Larry Wall stated that CQL aims to 'make simple queries easy and complex queries possible'. CQL forms part of the catalogue service specification in which CQL was standardised for OGC (OGC 2007a). In this standard, the Backus Normal Form (BNF) definition for OGC CQL is provided which is implemented into GeoServer.



## Chapter 4 Spatial data infrastructure and Geoportals

### 4.1. Introduction

A Spatial Data Infrastructure (SDI) is a basis for the sharing and exchange of spatial data. The entry point of an SDI is a geoportal that allows the user to search, discover, view, and edit data in an SDI. The functionality of geoportals are provided through web service. This chapter provide an overview of SDIs, geoportal and the evolution. The ThematicWS service needs to be designed so that it can be incorporated into a SDIs' geoportal, and thus understanding how an SDI and geoportal work is required for the development and implementation of ThematicWS.

### 4.2. Spatial Data Infrastructure (SDI)

#### 4.2.1. Overview

The spatial data infrastructure (SDI) emerged in the early 1990s when the emphasis moved from a stand-alone GIS towards a networked and collaborative information infrastructure (Nedović-Budić et al., 2011), and coincided with the boom of the World Wide Web (WWW). The SDI cookbook (GSDI 2004) defined an SDI as the basis for spatial data discovery, evaluation, and application for users and providers within all levels of government, the commercial sector, the non-profit sector, academia and by citizens in general. Rajabifard et al., (2006) describes an SDI as a platform which facilitates and coordinates the exchange and sharing of spatial data between the stakeholders. Masser (2005) stated that an SDI supports ready access geographic information. This is achieved with standards and effective mechanisms for the development and availability of interoperable geographic information and relevant technologies to assist in decision making for all stakeholders. In the World Bank SDI report (2011), an SDI facilitates the discovery of and access to harmonised spatial data through a collection of technologies, policies and institutional agreements. Coetzee (2011) has stated that an SDI is a collection of technologies, systems (hardware and software), standards, policies, agreements, human and economic resources, institutions and organisational aspects that have to be orchestrated to be possible for users to access data.

From the above definitions it can be seen that an SDI is defined in numerous ways, but have the following in common: an SDI is an network-based GIS which should provide reliable and effective access to geographic information and the relevant web services. Data interoperability is the main focus of SDI; datasets are harmonised so that integration of individual datasets is possible. The aim of the SDI initiative is to share and make data available to a wide audience for their economic and societal benefit. The access point of an SDI is usually an internet-based platform called a geoportal. A geoportal is used since the data inside an SDI is distributed and an internet-based platform makes it easier to search and find data relevant to a specific project or topic (Craglia, 2010). An SDI encourages the development of powerful and flexible data services for the geospatial computation and

analysis (Huang, Li & Zeng, 2007). Van Loenen et al., (2009) states that the focus of SDIs has moved from a data oriented (1990), to process oriented (1990 – 2005) towards a service-oriented SDIs demonstrated by INSPIRE.

#### 4.2.2. SDI components and hierarchy

An SDI is a combination of components: geographic data, metadata, frameworks, services, clearinghouses, standards, partnerships, education and communication (World Bank, 2011). These components are all needed to ensure sharing and access to data across multiple private and public organisations. The collaboration between stakeholders reduces the risk for any single organisation as they now share the risk, resources and responsibilities.

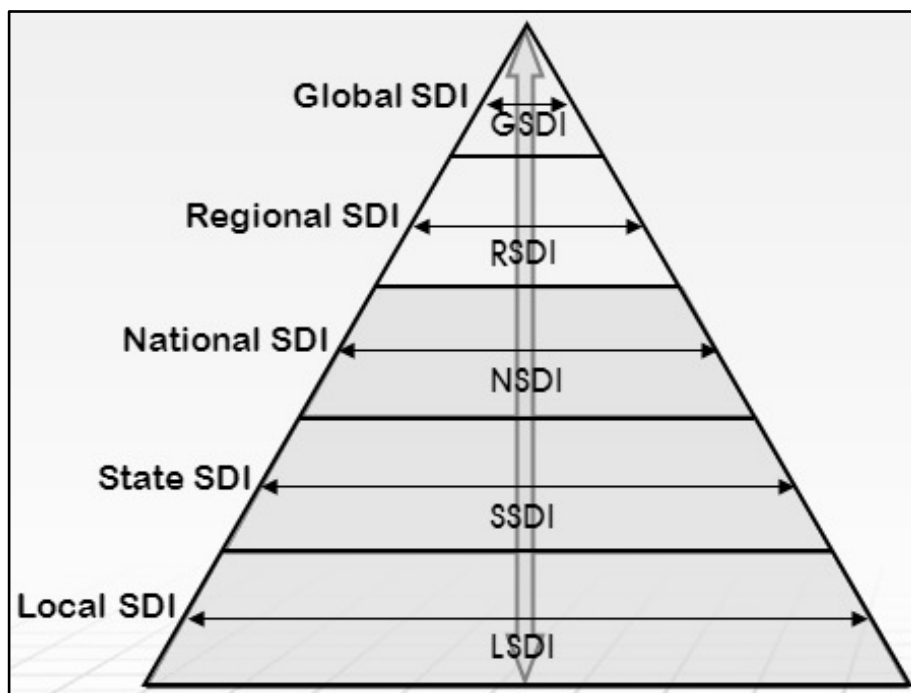


Figure 22. SDI hierarchy (Rajabifard et al., 2006, World Bank, 2011)

An SDI consists of inter-connected SDIs, as shown in Figure 22. Rajabifard et al., (2006) state that the SDI hierarchy allows decision makers to access and share geographic information from any other level in the hierarchy. The themes, scales and coverage of the data depend on the level of the SDI hierarchy the data is accessed from. Figure 22 also shows that the GSDI can be developed top-down or bottom up.

#### 4.2.3. Examples of SDI initiatives

In 2010, there were a recorded number of 105 countries that have established a national spatial clearinghouse, which is a 25% increase from 2005 (Nedović-Budić et al., 2011). Currently, the majority of SDIs are located in America and Europe where 60% of the countries have established a national clearinghouse, compared to only 20% in Africa.

There is a number of successful SDI worldwide: **IN**frastructure for **SP**atial **InfoR**mation in **E**urope (INSPIRE) was launched by the European Commission in 2001. INSPIRE is implemented by the 27 member states of the European Union (World Bank, 2011). The aim of INSPIRE is to make relevant data available and accessible to support environmental and social policies.

The Netherlands started the development of a national SDI (NSDI) in 1992, which was designed by the National Foundation for Geographical Information (World Bank, 2011). The increase of interoperability and exchange of geographic information was the original goal of the NSDI. As part of the development process of the Dutch SDI, the fundamental datasets were identified to aid in the decision making of all stakeholders. The geo-standards Wiki is an initiative by the Dutch SDI to aid in the understanding of the standards required for the development of SDI.

Australian and New Zealand Land Information Council implemented the Australian Spatial Data Infrastructure (ASDI) in 2003 for providing users with spatial information (World Bank, 2011). ASDI enables the easy and cost effective access to geographic information and services to all stakeholders, which includes a wide range of organisations in the public and private sectors in Australia and New Zealand.

The United States (US) National Spatial Data Infrastructure (NSDI) was initiated in 1994. The Federal Geographic Data Committee (FGDC) is responsible for the implementation of the NSDI. The implementation has two significant parts: Geospatial One-Stop (GOS) and The National Map (TNM). GOS is the geoportal that facilitates the data sharing and exchange. Most states are actively involved in the development and maintenance of the NSDI. The US government has acknowledged the benefits of the SDI.

The Canadian Geospatial Data Infrastructure (CGDI) focuses on four main themes: public safety, public health, environment and aboriginal affairs (World Bank, 2011). CGDI has over 1400 databases in its network that can be accessed and shared over the Internet. CGDI consider international standards very important, rather than creating national standards. This enables the CGDI to be easily integrated into a global SDI.

The previously mentioned SDIs are some of the most successful SDI initiatives worldwide. In Africa, the experience is a bit more diverse. The initiatives have started out overly ambitious, complex, and fragmented, with too many competing demands (World Bank, 2011); however, they have lost traction.

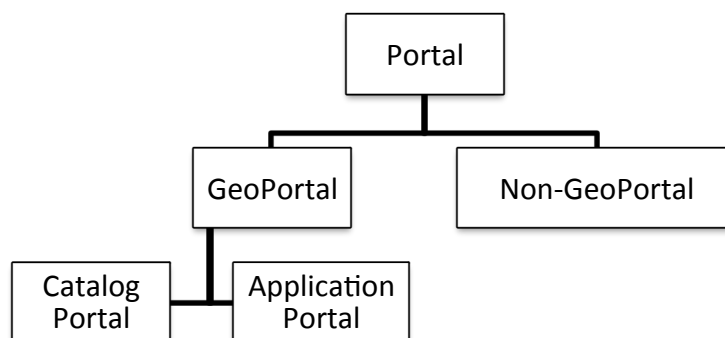
The South African Spatial Data Infrastructure Act of 2003 established the South African Spatial Data Infrastructure (SASDI) and the Committee for Spatial Information (CSI). According to Clarke (2011), the vision of SASDI is to aid stakeholders in making appropriate decisions in the development process through the effective use of spatial data for the benefit of all South Africans and the environment. The strategic objective of SASDI can be divided into two parts: *access to* and *use of* the spatial data in the spatial data infrastructure (SDI). The *access to* part will consist of the setup of the SDI, gathering the data, cataloguing, etc. Thereafter, the *use of* part will commence, getting people to use the SASDI data and services.

### 4.3. Geoportals

Tait (2005) defines a portal as a web site, which is an entry point to other web locations. Tait deduced from this definition that a geoportal is thus a web site that presents an entry point to geographic content on the web or more simply a web site where geographic content can be discovered. However, this definition is too general; it includes web sites with an address location and map and in these types of web sites the geographic content is supported rather than being the main focus. A geoportal is redefined as a web site where the discovery of geographic information is the primary focus. A geoportal is a World Wide Web (WWW) gateway that organises geographic information and the relevant services (De Longueville, 2010; Maguire & Longley, 2005). He et al., (2011) states that a geoportal provides access points to different sources of geographic information on the web.

A geoportal is the most visible part of an SDI, and provides access to spatial data and associated web services for discovery, display, editing and analysis. An SDI plays an important role in the sharing of geographic information and aims to avoid duplicated efforts, inconsistent data and wasted efforts. The geoportal thus has a major role in the simplifying of contributing and accessing spatial data available in the SDI (De Longueville, 2010). The focus of a geoportal is interoperability, which is obtained through the implementation of standards for the discovery and use of spatial data and services. Geoportals can be considered as a kind of mash-up, since a mash-up is a web application that combines data from multiple sources into a single integrated tool.

Figure 23 depicts the classification of geoportals as described by Maguire and Longley (2005) into two types. A catalog geoportal is concerned mainly with the organising and management of the access to geographic information. An application portal provides readily accessible to online spatial web service.



**Figure 23. Classification of web portals**

The INSPIRE state of play reports (2011) that most countries in the EU have developed or are currently developing a geoportal, either at national or provincial geoportals, as an access point to INSPIRE. Most of the geoportals are very successful, for example the French geoportal. The geoportal has an estimated number of 3 million unique visitors per year. Spain, Poland and the Czech

Republic's geoportals average between 300 000 and 500 000 unique visitors per year. Other national geoportals, however, do not exceed 20 000 visitors.

With the increasing popularity of geoportals, the evaluation of their usability has become a topic of interest. He et al., (2011) developed a method for evaluating usability of geoportals based on ISO 9241-11 and ISO TC159/SC4/WG11 framework. Effectiveness, efficiency and user satisfaction are the principal factors to consider. The Swedish national geoportal was evaluated and the result of this geoportals' usability was fairly good and additional weaknesses have been identified.

#### **4.4. Moving towards an intelligent geoportal**

The formal definition of data is reinterpretable representation of information in a formalised manner suitable for communication, interpretation, or processing (ISO/IEC 2382-1). A more informal description of data is raw, unorganised fact. Information is knowledge concerning objects such as facts, events, things, processes, or ideas, including a concept that within a certain context has a particular meaning (ISO/IEC 2382-1). Information is produced when data is processed, organised and structured in a given context so as to make it useful.

Spatial information infrastructure (SII) is used interchangeably with a spatial data infrastructure (SDI); however, a spatial information infrastructure is more accurately the next step in the evolution of the traditional SDI (Iwaniak & Kubik, 2010; Rautenbach et al., 2012a). An SII provides access to *information*, which requires intelligence to orchestrate (automatically coordinate) web services that prepare and present information, instead of data, to the user. The access point to the SII would be an *intelligent geoportal*, a geoportal that provides complex functionality through a simple user interface for a user in a specific application domain (Iwaniak et al., 2011). In an SII, the focus moves from interoperability of data to interoperability of *processed data*, which makes it possible to search, retrieve and present information.

## Chapter 5 Related work

### 5.1. Introduction

Thematic mapping has become an increasingly popular topic in recent years. Interest has especially been shown in the use of web services to produce thematic maps; this can be attributed to the rapid development in SDI and web service technologies. Even with recent developments in web service technologies like SLD and SE, there still lacks specific functionalities for the produced thematic maps. At present, there is no open source solution for the production of thematic maps using web services; thus, there is no explicit OGC standard.

There has been a couple of suggestions in recent years that the main focus has been on extending SLD and SE but there has also been a suggestion to modify KML for thematic mapping. This section provides an overview of these suggestions.

An overview of related work in the fields of thematic mapping using OGC standards, WPS implementations and issues, and the orchestration of OGC web services are provided. Current challenges for the development of a ThematicWS are discussed.

### 5.2. SLD experiments and implementations

#### 5.2.1. Extending SLD and SE for cartograms

Rita et al., (2010) present a proposal for an extension of SLD and SE, which allows the defining of cartograms. SLD and SE are used, since these standards are responsible for defining the representation of objects on the map. A layer can be distorted according to a selected numeric attribute selected with this extension.

The Extension was implemented in GeoServer and uses the ScapeToad library to produce the cartograms. The new symbolizer is called the CartogramSymbolizer, which is placed in the rule element of the SLD style sheet. The CartogramSymbolizer element consists of two subtypes: PropertyName and DistortionQuality. The PropertyName specifies the numeric attribute, which will be used to deform the layer, and DistortionQuality defines the quality of the resulting cartograms. The extension implements the Gastner and Newman diffusion algorithm for the producing cartograms.

#### 5.2.2. Thematic extension for SLD: SLD-T

Sae-Tang and Ertz (2007) proposed an extension to the SLD and SE standard called SLD-T. Their method focuses on choropleth and proportional symbol maps. In the past, Ertz has spoken out about the limitations of SE and SLD, especially with regards to thematic cartography.

Producing thematic maps with SLD can become complex and unpleasant with each filter needing to be within a rule in the SLD document. SLD-T was developed to address these problems and aims to

reduce redundancy and verbosity in the standard SLD. Each rule element is extended with a thematic symbolizer in SLD-T. The following new symbolizers are available in SLD-T:

1. **CategoryThematicSymbolizer**  
The **CategoryThematicSymbolizer** is used for classified maps. This symbolizer is built-on **ThematicCategory** elements to describe the classification type.
2. **SimpleThematicSymbolizer**  
The **SimpleThematicSymbolizer** is used to produce unclassified maps. This symbolizer is a simple wrapper of the standard symbolizers that allows them to inherit useful generic elements, like symbol priority and placement.
3. **MultiThematicSymbolizer**  
The **MultiThematicSymbolizer** is used to produce thematic maps that combine different techniques, for example combining choropleth and proportional symbol maps.
4. **ChartThematicSymbolizer**  
The **ChartThematicSymbolizer** is used to produce diagram maps.

### **5.2.3. Thematic Symbology Encoding (TSE)**

Dietze and Alexander (2007) proposed an extension to SLD and SE, called Thematic Symbology Encoding (TSE). This extension allows the definition of choropleth and diagram maps. The extension is similar to the 3D extension of SE, in which an XML schema is defined for this purpose. Similar to the other methods the extensions are embedded in the rule element. In this method a new namespace is introduced called *tse*.

The **ThematicSymbolizer** defines how a feature should appear on a map and provides additional cartographic thematic elements to a polygon. The **DiagramSymbolizer** enables the user to draw shapes or surfaces in the form of small diagrams or graphs. Both these symbolizers have some subtypes that allow the specification of properties.

SLD is a very powerful standard but has its limitations, especially due to the fact that only predefined symbols can be used. This problem was overcome by producing the TSE specification. This proposal was used successfully in client-side thematic map production, and in some web service implementations.

### **5.2.4. KML and geobrowsers for thematic mapping**

The previous methods were all similar in the fact that they all suggest an extension to the SLD and SE standards. Sandvik (2008) suggested a different approach using KML and Geobrowsers. Geobrowser has become a very popular tool for viewing spatial data, especially for novice users. The other main component of the suggested solution is KML, which is not targeted towards thematic mapping, but it is possible to manipulate the current KML elements. KML and Geobrowsers offer great potential for thematic mapping; however, there are still some issues to be resolved.

The implementation of this solution uses an easy-to-use web interface to produce the thematic maps and Google Earth as the browser to display the map. The method was implemented to be able to produce proportional symbol, chart and choropleth maps. KML supports limited styling, thus, this needs to be added or the use of SLD must be incorporated in KML. The Google Chart API was also utilized to produce chart maps. Figure 24 provides a snapshot of how the KML can be extended to include styling. This method is also examined in Section 6.2.3.2.

```

1  <kml>
2    <Document>
3      <Style id='sharedStyle'>
4        <IconStyle>
5          <Icon>
6            <href>files/symbol.png</href>
7          </Icon>
8        </IconStyle>
9      <Style>
10     <Folder>
11       <Placemark>
12         <name>China</name>
13         <Snippet>1,312,978,855 (2005)</Snippet>
14         <styleUrl>#sharedStyle</styleUrl>
15         <Style>
16           <IconStyle>
17             <color>e50066ff</color>
18             <scale>7</scale>
19           </IconStyle>
20         </Style>
21         <Point>
22           <coordinates>106.514,33.421,0</coordinates>
23         </Point>
24       </Placemark>
25     </Folder>
26   </Document>
27 </kml>

```

**Figure 24. KML code use for thematic mapping, from the KML for Thematic Mapping (Sandvik, 2009)**

### 5.2.5. Web-based thematic map service in OpenGIS environment

Hong and Lin (2005) proposed a Thematic Map Service that is entirely based on OpenGIS technology. Figure 25 shows the proposed architecture of the system. Thematic Map Server serves as a bridge between the client and the other services. The Geolinked Data Access Service (GDAS) server stores the data in an XML file and the Geographic Linkage Service (GLS) server builds a link between the attribute data acquired from the GDAS and the corresponding map. The WMS server creates the thematic map with SLD. What makes this proposal different is the fact that the service has some built-in knowledge, which helps to ensure that a high quality thematic map is produced.



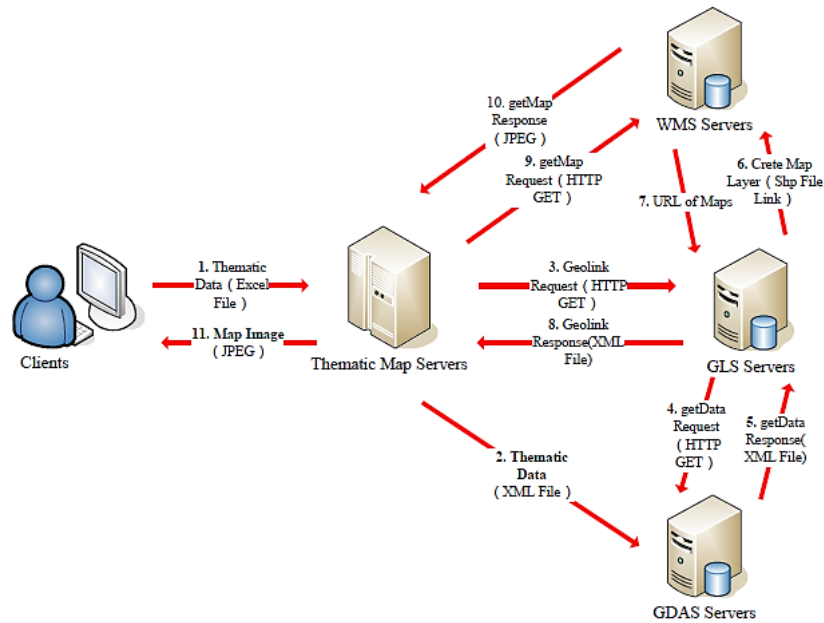


Figure 25. 3-Tier system architecture as well as the process of the thematic map service (Hong & Lin, 2005)

### 5.2.6. XSLT template for thematic maps

Extensible style sheet Language Transformations (XSLT) is a declarative XML-based language used to transform XML documents, which is standardised by the W3C. XSLT was selected for the following reasons (Čerba, 2010):

1. Compatible with XML, GML and SVG standards;
2. Strongly supported by numerous software packages; and
3. Applications created with XSLT are multiplatform, modular and well structured (refer to Figure 26).

To produce a thematic map, three types of XML are needed: coding and describing source geographic information, the second is for coding and describing the thematic map and the last one is for the transformation and styling.

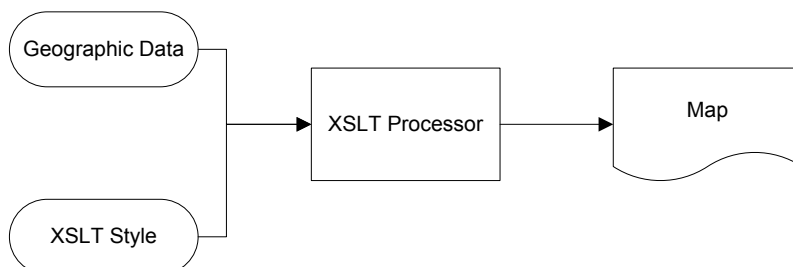


Figure 26. Basic Process of the XSLT Template approach (Čerba, 2010)

## **5.3. WPS research**

### **5.3.1. Availability of OGC WPS services**

Lopez-Pellicer et al., (2011) conducted a survey in March 2011 on the availability of services, which conform to the OGC WPS standard. The services were discovered using metadata records on the WPS servers in spatial catalogues available online. They found that there are 9 329 OGC services available online, confirming the OGC implementation statistics (OGC 2011), which state that the WMS specification is the most implemented of the OGC web service standards, with 46.1% of services discovered.

Only 58 WPS services were discovered, with 47 implementation of the WPS 1.0.0 service and 11 WPS 0.4.0 implementations. The WPS services discovered offered 1 316 processes, more than half of the services offering four or less processes. Universities were found to be the top service provider and 84% located in Europe. In WPS process description information, mainly the mandatory elements were provided, identifier and title. The optional elements, such as processVersion, profile and WSDL was found in less than 5% of the discovered services. The DescribeProcess request was successful in 92.4% of the services. The survey identified the following barriers for the technical and semantic interoperability needed for the orchestration of web services: the lack of documentation, profiles and few WSDL descriptions available. From these results it can be deduced that WPS implementations are still an academic exercise and that the service is not being used by the industry.

### **5.3.2. Compliance of WPS services**

At the FOSS4G 2011 conference, Garnett and Fenoy presented a WPS shootout; they tested the frameworks compliance to the OGC WPS standards and interoperability of the framework. The following five WPS frameworks were evaluated: 52° North, constellation, deegree, GeoServer, PyWPS and ZOO project. The authors were aided by experts from the different frameworks and results hosted by the ZOO projects website.

All the evaluated frameworks, except the constellation project, passed the tests set out in the shootout. As for the constellation framework, none of the tests could be performed as the framework is still under construction.

## **5.4. Web service implementations and orchestration**

### **5.4.1. Cost of implementing web services in an SDI**

Web services are regarded as the main technological drivers of the second generation SDIs, because they can fulfil the needs of users and improve the use of data. However, there are costs involved in every phase of the web service life cycle such as setting up and maintaining web services, and the hardware, software, and expertise required to develop web services (Donker, 2009). A study looked into the cost recovery models presently employed in INSPIRE, and whether or not they are sufficient. Donker (2009) found that currently these revenue models do not cover all costs involved in web

service development and maintenance. It was also stated that in the long-term, making web services available free of charge or for a low cost will have intangible benefits and more value added products will be created, justifying the cost involved in using web services.

#### **5.4.2. Use of BPEL for the orchestration of OGC web services**

Business process execution language (BPEL) is the *de facto* for orchestration of processes. BPEL4WS was created specifically for the orchestration of web services; however, researchers have shown that BPEL has a number of limitations when orchestrating OGC web services (Weiser & Zipf, 2007; Stollberg & Zipf, 2007; Schaeffer 2008; Fleuren & Muller, 2008).

The main obstacles identified at the time were the lack of SOAP support, raw binary data served by OGC services, and the manual creation of the WSDL document for each process. These make it difficult to use BPEL engines for the orchestration of OGC web services. The SOAP support problem has been addressed by OGC, by adding SOAP support to all of their web service specifications.

Fleuren and Muller (2008) suggest using a PostGIS database to store intermediate data for orchestration with an ActiveBPEL engine as a solution to the raw data transfer problem. In some other BPEL engines, data managements have been added; the data returned by the web services are copied into a variable, so that the necessary modifications can be made. However, when the data volumes reach terabytes, a multiple workflows executes simultaneously then bottlenecks and other problems can occur (Fleuren et al., 2010).

Wrapping existing geoprocessing functionality in a WPS makes it possible to deploy these services in the cloud, which is especially useful for resource intensive computations (Ludwig & Coetzee, 2010). Stollberg and Zipf (2007) investigated the use of WPS for the orchestration of OGC web services as an alternative for BPEL. One approach involved using a WPS service as a central controller that activates the services. Another composed all the processes into a single WPS service. All approaches show that a WPS service can be successfully used to orchestrate web services.

#### **5.4.3. Feature Portrayal Service (FPS)**

As an alternative to the ThematicWS service, the Feature Portrayal Service (FPS) has been suggested. FPS is an OGC service specification that applies a style to feature data in order to produce a map (OGC 2005). Feature data retrieved from a WFS request can be styled using an identified (via a URL) or specified style by the client. However, the specification of the FPS has not been updated since 2005 and the document is still in draft format (version 0.0.30) with visible comments in the document. There are also at this time no implementations available of the service, as shown in the OGC implementation statistics (OGC 2011).

## 5.5. Semantic web services

The development of the World Wide Web (WWW) allowed computers to understand and display web pages created using HyperText Markup Language (HTML), without having access to the intended meaning. The semantic web is an extension of the Web, which adds a new machine-understandable metadata layer that enables the processing of data and information over the Web (Cardoso 2006). In order to rely on the concept of the semantic web, the W3C have developed open standards, such as the Resource Description Framework (RDF) and Web Ontology Language (OWL). These standards are essential for integration and interoperability of data and services. Semantic web services have emerged from on-going research and evolution of web services (the syntactical definition), and the concept of the semantic web.

Semantic web services have been developed using the following approaches (Cardoso 2006, Zaharia *et al.* 2008, Yue *et al.* 2007): mapping the concepts in WSDL to ontological concepts, describing web services semantically using OWL-S, and using WSMO to provide ontological specifications.

Ontologies have been the latest buzzword in numerous research and application fields. Stock *et al.* (2012) investigated the usefulness of ontologies in a knowledge infrastructure. Within the knowledge infrastructure the benefits of using ontologies for resource discovery were investigated. It was determined that a geospatial knowledge infrastructure can be enriched with ontologies (semantics) and registry standards (interoperability). Stock *et al.* recommends that all components be stored using ontologies so that the opportunity for semantic discovery and inference are maximised.

Zaharia *et al.* (2008) developed a platform named SWING (Semantic Web Service Interoperability for Geospatial web services) as part of a project to increase the level of automation for all web service aspects. WSMO was adopted for the project to provide a more efficient solution for the automated execution of spatial queries using semantically described services. Janowicz *et al.* (2010) proposed adding a semantic layer (Semantic Enablement Layer) with a Web Ontology Service and Web Reasoning Service to enable the semantic discovery, dynamic orchestration of services and interoperability to a SDI.

## Chapter 6 Modelling the thematic cartography process

### 6.1. Introduction

In Section 2.7, the theoretical thematic cartographic process for choropleth and proportional symbol maps is described. However, in this chapter an evaluation of visualisation standards, thematic mapping applications (desktop and online), and application programming interfaces (API) is performed to understand how the theory discussed is implemented in these applications. The following aspects of the applications are evaluated: implemented thematic cartography methods, classification and data standardisation methods available, and the visualisation options. Based on the evaluation performed, a model of the thematic cartography process is produced. The model produced is depicted as a flow diagram, which provides the framework for the design of the ThematicWS conceptual model.

### 6.2. Evaluation of thematic cartography implementations

#### 6.2.1. Evaluation criteria

For the evaluation of the thematic cartography implementations, the following were examined:

1. Thematic cartography methods implemented.
2. Classification methods available.
3. Data standardisation methods available.
4. Visualisation options.

Section 6.2.2 and 6.2.3 provide a review of the implementation and a description of the implementation to produce choropleth and proportional symbol maps. Even though the project focuses on choropleth and proportional symbol maps, the first criteria evaluates all the thematic methods in order to deduce the most popular implementations to support the decision of implemented methods in ThematicWS.

#### 6.2.2. Desktop GIS applications

##### 6.2.2.1. ArcMap

ArcGIS is a proprietary GIS software suite developed by ESRI for the Windows operating system. ArcGIS aims to improve workflow and solve challenging issues in numerous industries with regards to spatial information. The functionalities included in the suite are asset and data management, planning and analysis, business operations and situational awareness (ESRI 2010). ArcGIS 9.3 was used for this evaluation.

Thematic maps can be produced on a single map layer on one or multiple attributes depending on the type of thematic map selected. The thematic options can be accessed through the layer properties and then symbology. Figure 27 is the input form to produce an unclassified choropleth map. The inputs

required from the user are the attribute according to which the thematic maps must be produced, and the colour ramp. ArcGIS provides the users with predefined colour ramps that are aesthetically pleasing and optimal for the user to select from.

As for classed choropleth maps, the attribute, colour ramp and classes must be specified (refer to Figure 28). However, the classes have an extra view, under classification, that provides the different methods of classifications. In this view, it provides the necessary statistics and a histogram (Figure 29) that shows how different classification methods change the group distribution.

For proportional symbols, the following inputs are required: the attribute, colour and shape of the symbol, background colour, and the number of symbols to be displayed on the legend which must be selected (refer to Figure 30). The maximum size of the symbols cannot be specified; this is calculated by the application. The overall symbology of the maps can be easily changed and customised by the user through the styling interface. The resulting choropleth and proportional symbol maps are displayed in Figure 31.

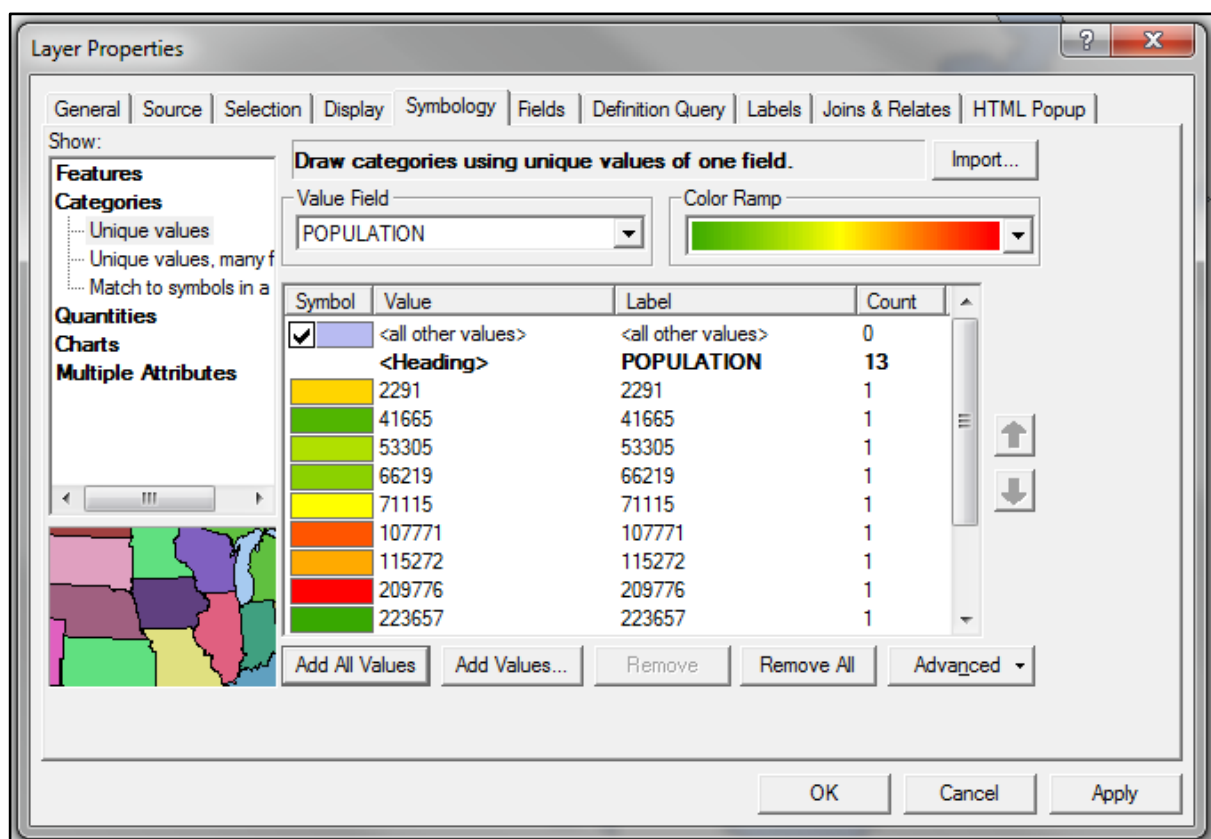


Figure 27. ArcMap screen to produce an unclassed choropleth map

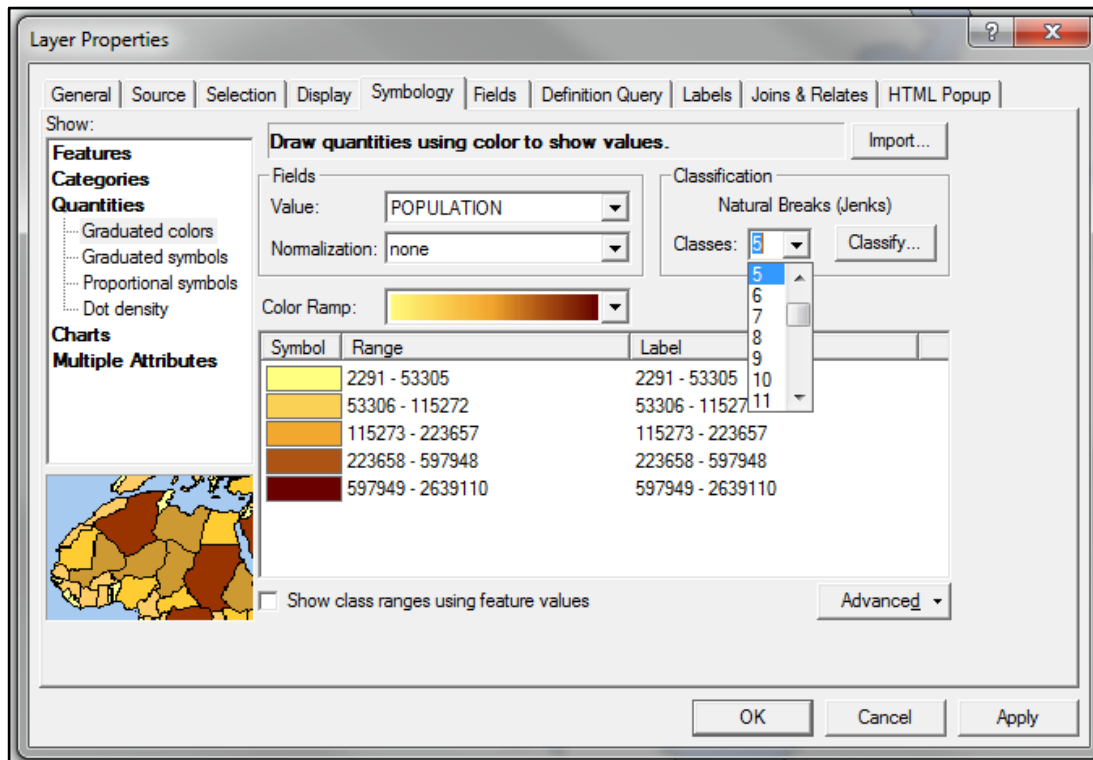


Figure 28. ArcMap screen to produce a classed choropleth map

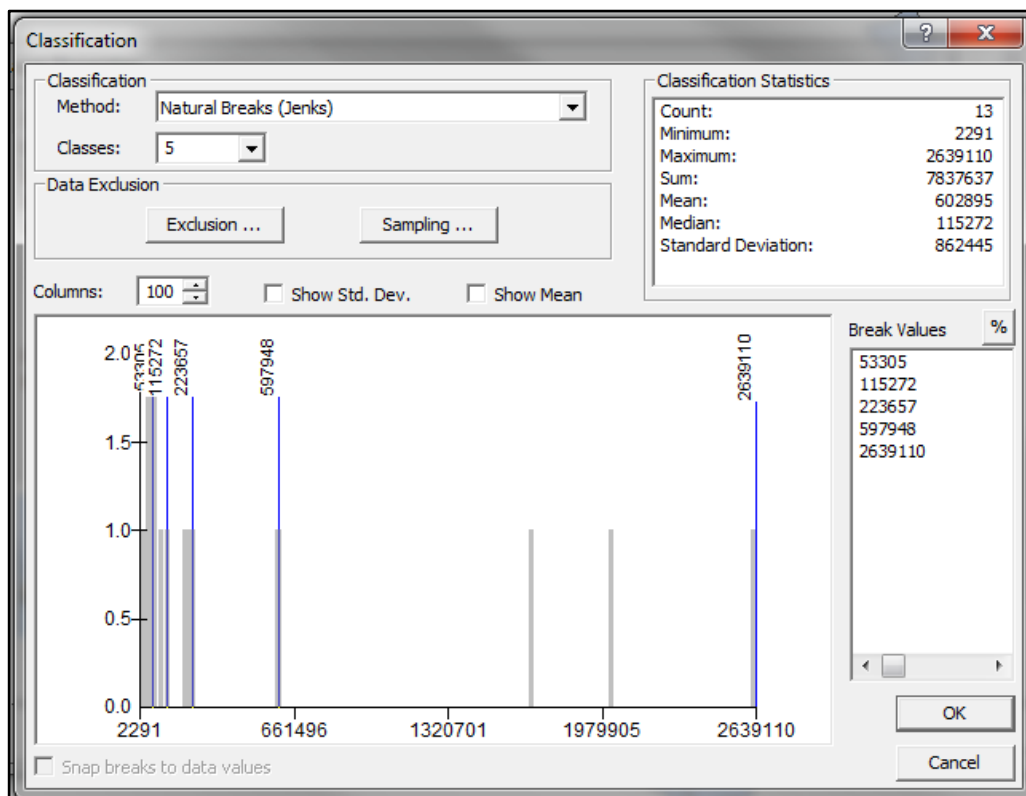


Figure 29. Histogram to evaluate the different classification methods

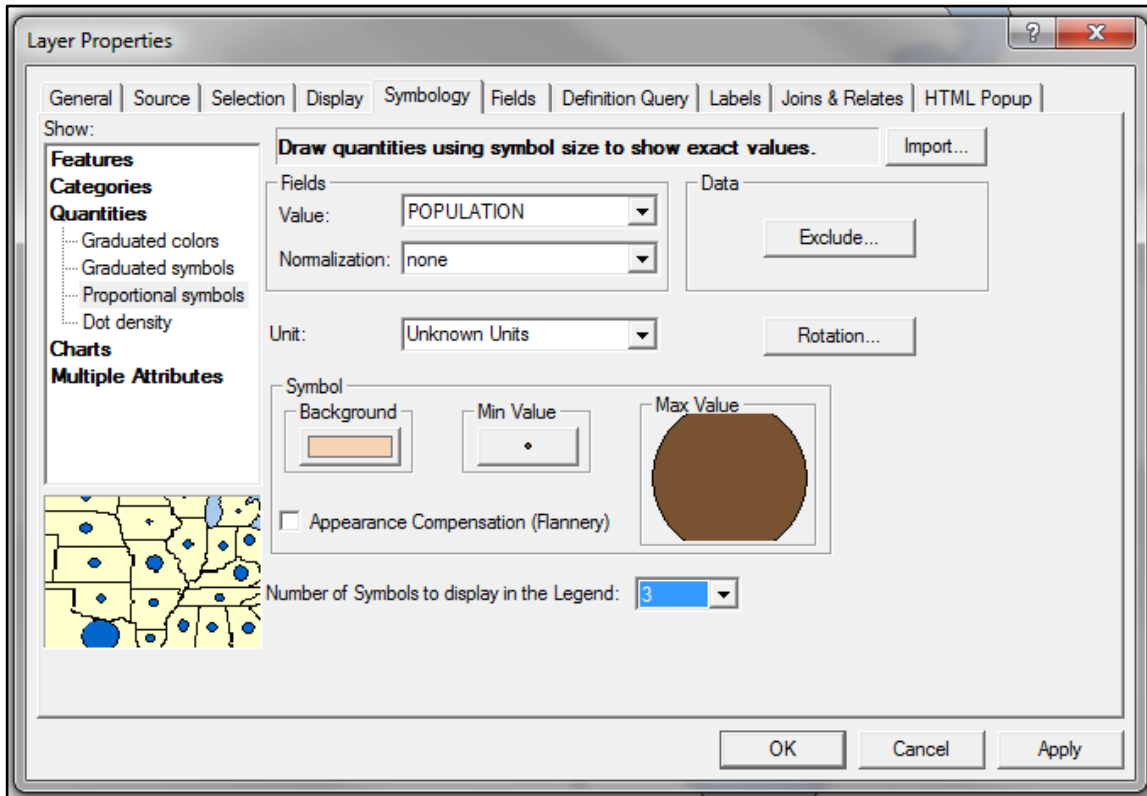
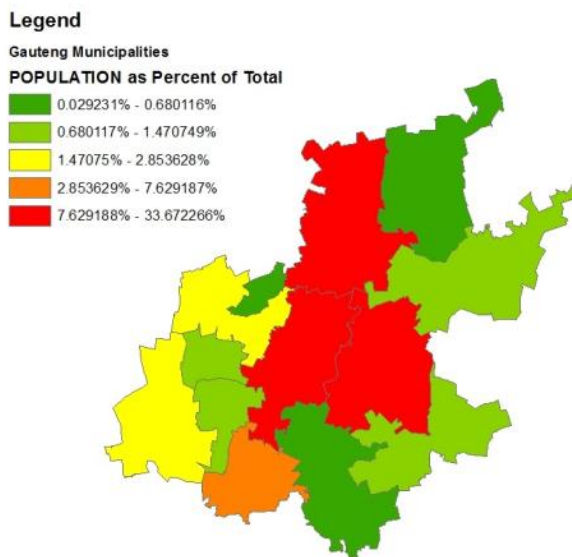
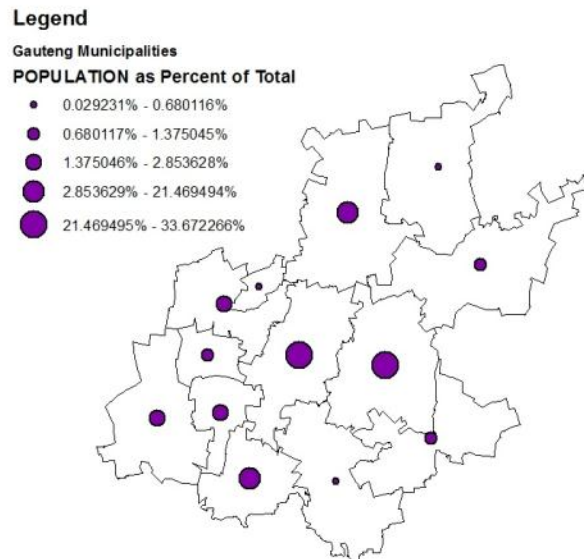


Figure 30. ArcMap screen to produce a proportional symbol map



a) Choropleth map with five classes



b) Proportional symbol map

Figure 31. Examples of thematic maps generated with ArcGIS



### 6.2.2.2. Quantum GIS

Quantum GIS (QGIS) is an open source, cross-platform GIS program. QGIS is an application that provides data viewing, editing, and analysis capabilities for the following formats: vector, raster, and database (<http://www.qgis.org/>). QGIS is open source software, meaning that the software source code can be freely viewed and modified. The General Public License (GPL) places a restriction that any modifications made to the source code must be made available to the QGIS project and that a new version of QGIS cannot be created under a 'closed source' license. QGIS version 1.6 was used for this evaluation.

The QGIS interface differs slightly from ArcMap, but the input needed is similar. The following needs to be specified for choropleth map: the attribute, method of classification, number of classes and the colouring specifications. The colouring in QGIS is not as rigid as in ArcMap since the user can decide which colours to use, which may cause some aesthetic problems. Figure 32 show the user interface in QGIS for producing a choropleth map.

As for a proportional symbol map, the requirements are: classification type, classification attribute, symbol, and symbol size (refer to Figure 33). Custom symbols can be uploaded and used in the proportional symbol map.

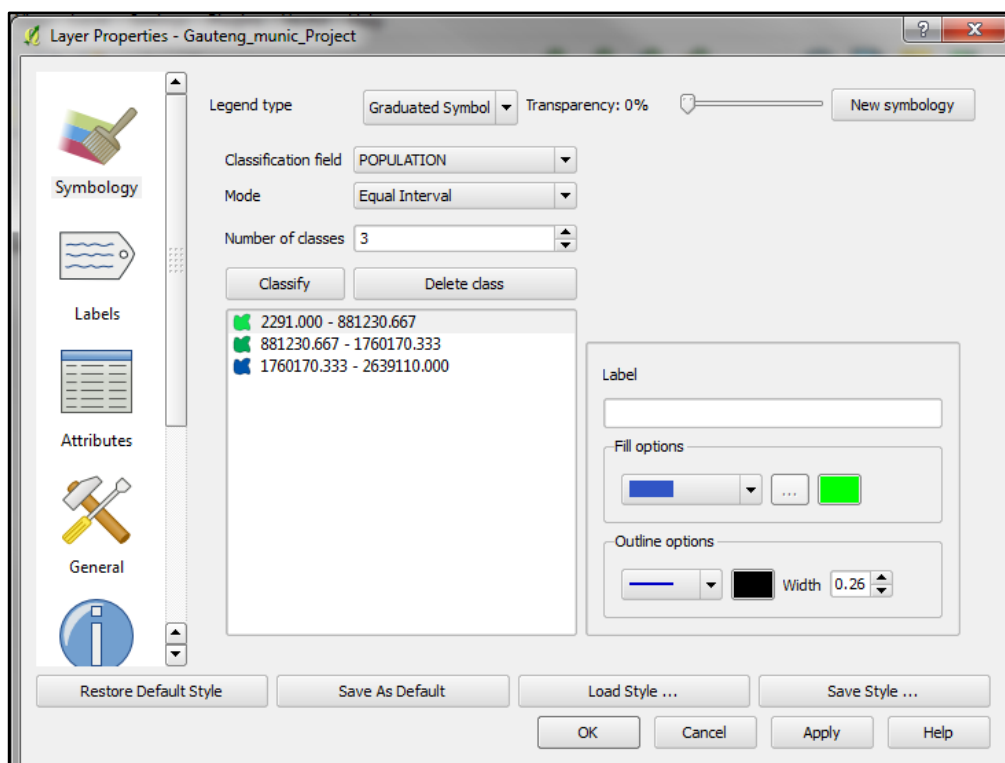
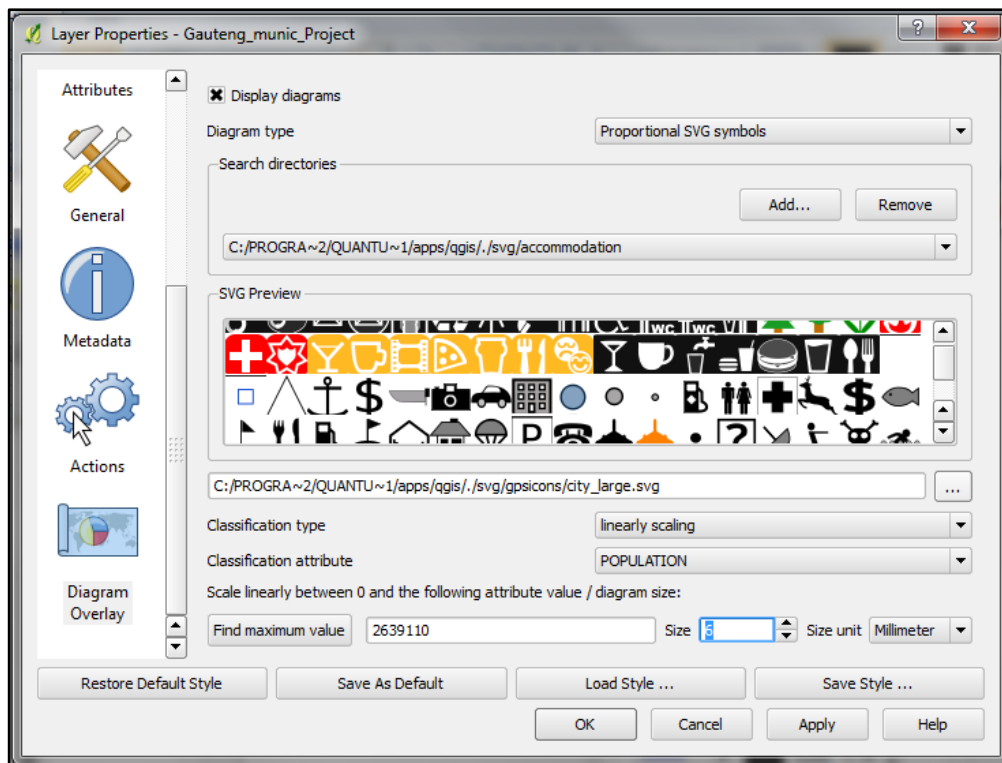


Figure 32. QGIS user interface for producing a classed choropleth map



**Figure 33. QGIS user interface for producing proportional symbol maps**

### 6.2.2.3. *OpenGeoDa*

OpenGeoDa is a free-ware software package specialising in spatial analysis, geovisualisation, spatial autocorrelation and spatial modelling. GeoDa has powerful capabilities to perform spatial analysis, multivariate exploratory data analysis, and global and local spatial autocorrelation (Centre for Spatially Integrated Social Science, 2003). GeoDa does not specialise in symbolising but rather in data exploration.

GeoDa does not have the ability to produce proportional symbol maps; for this software only choropleth maps were examined. Through the Map Menu (refer to Figure 34 and 35), thematic maps can be produced. It is not possible to produce unclassed maps with GeoDa. After the classification method is selected, a dialog box appears prompting the user to select the attribute followed by a dialog for the number of classes, and a choropleth map is produced as a result. GeoDa determines the colouring of the classes and the user cannot alter it.

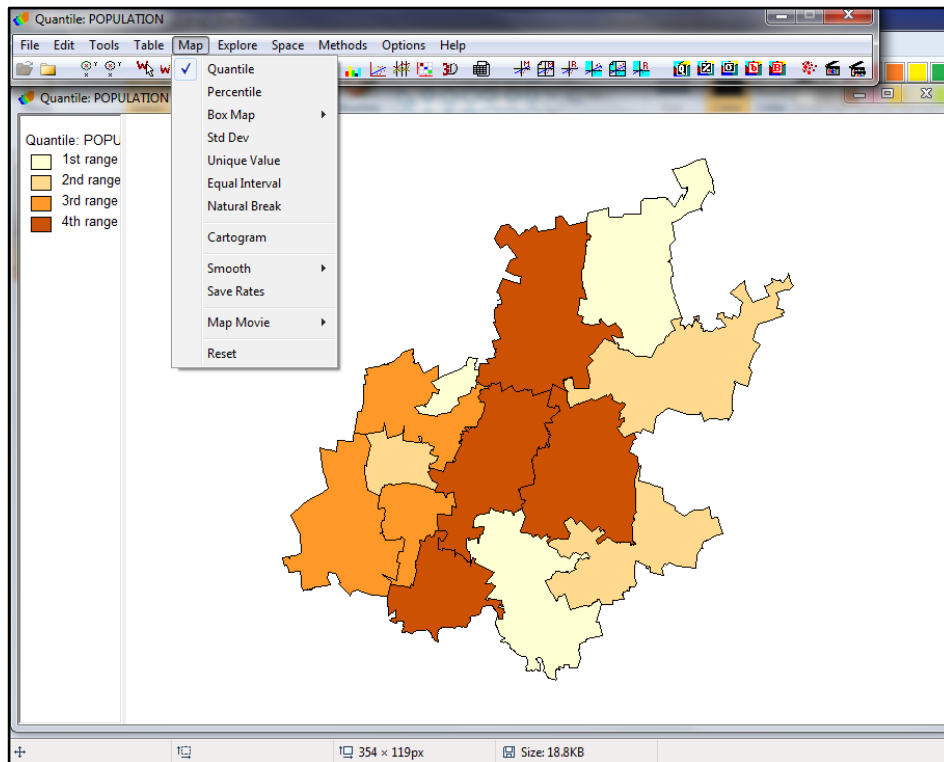


Figure 34. GeoDa, the produced choropleth map and the different types of classification can be seen

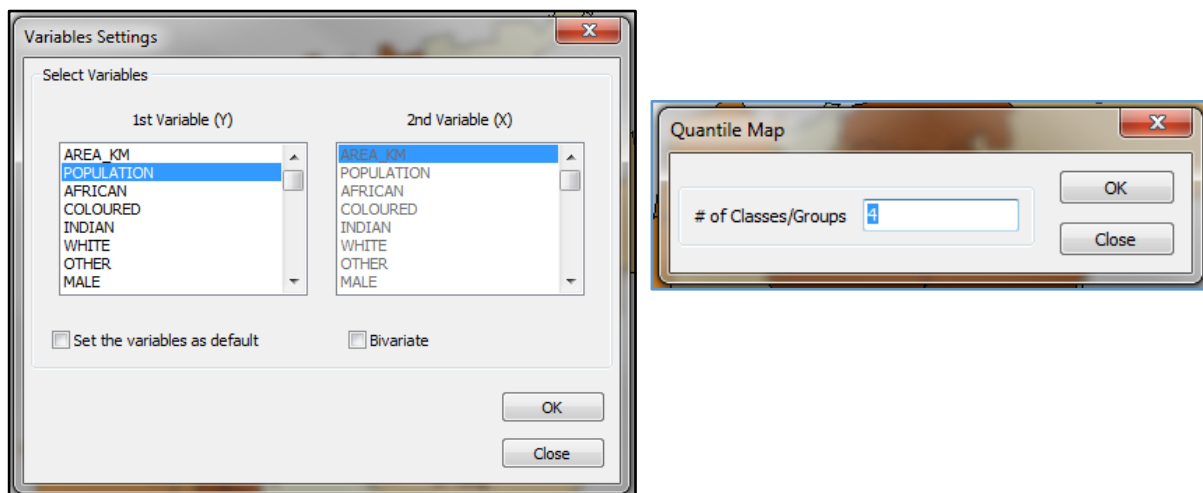


Figure 35. GeoDa dialog for selecting the attribute and number of classes

## 6.2.3. Web applications and Thematic Mapping APIs

### 6.2.3.1. Standard SLD and SLD extensions in GeoServer

Styled Layer Descriptor (SLD) is an Open Geospatial Consortium (OGC) standard for defining visualisation styles that can be used for publishing raster and vector data on the Internet. Standard

SLD is described in Section 3.5, with examples of style sheets to produce choropleth and proportional symbol maps.

GeoServer implements an enhanced implementation of the SLD standard, with extensions such as dynamic symbolizers, parameter substitution, and the Google charts extension to produce diagram maps. An overview with example is provided in Section 3.5.3.3.

### 6.2.3.2. *The Thematic Mapping API and engine*

The thematic mapping API was created as part of a research project to investigate and demonstrate how geobrowsers can be used to produce thematic maps (refer to Section 5.2.4). Thematic cartography has a long history, whereas the new geobrowsers technologies focus on detailed satellite imagery and general reference maps. Geobrowsers are very popular currently and, if it is combined with thematic mapping, it makes thematic cartography available to a wider audience.

**Table 3. Configuration options of the thematic engine (Sandvik, 2008)**

Configurations	Default	Description
alphaColour	220	
barSize	50000	
colour	'FFFF00'	
colourType	'single' for all map types except choropleth.	<ul style="list-style-type: none"> <li>• single</li> <li>• scale</li> </ul>
chartType	'pie'	<ul style="list-style-type: none"> <li>• pie</li> <li>• pie 3D</li> <li>• bar</li> <li>• column</li> </ul>
endColour	'FF6600'	
geometry	Null	JSON format
mapType	'choropleth'	<ul style="list-style-type: none"> <li>• choropleth</li> <li>• prism</li> <li>• chart</li> <li>• bar3D</li> <li>• symbol Image</li> </ul>
maxHeight	2000000	
startColour	'FFFF99'	
symbolHref		
symbolMaxSize	10	
symbolShape	'circle'	<ul style="list-style-type: none"> <li>• circle</li> <li>• square</li> <li>• triangle</li> <li>• octagon</li> </ul>

For more detail refer to Section 5.2.4. In this section, the parameters required to produce thematic maps are examined. The Thematic Mapping Engine produces thematic maps, which can be visualised in a geobrowser that supports KML (Sandvik, 2008). This API works together with the Google Visualisation API. The configuration options of the API are set out in Table 3 and can be set inside the JavaScript code (Figure 36 and 37) to produce thematic maps.

Figure 36 shows an example of the JavaScript code used to produce a choropleth map. Line 4 to 8 set the options set out in Table 3. However, the number of classes cannot be specified.

```
1   var data = response.getDataTable();
2   var map = new TME.Map.Kml.GoogleViz();
3
4   var options = {
5     type: 'choropleth',
6     title: 'Infant Mortality Rate',
7     classification: 'equal',
8     geometry: worldBorders
9   };
10
11  var kml = map.draw(data, options);
12  var kmlObject = earth.parseKml(kml);
13  earth.getFeatures().appendChild(kmlObject);
```

**Figure 36. Code to generate a choropleth map**

Figure 37 is the example of a proportional symbol map with line 4 to 11 used to set the options. The symbol can be chosen from a fixed list of symbols, for example circle and square.

```
1   var data = response.getDataTable();
2   var map = new TME.Map.Kml.GoogleViz( earth );
3
4   var options = {
5     type: 'symbolImage',
6     title: 'World Population',
7     symbolHref:
8     'http://thematicmapping.googlepages.com/circle.png',
9     symbolMaxSize: 8,
10    showBorder: true,
11    geometry: worldBorders
12  };
13
14  var kml = map.draw(data, options);
15  var kmlObject = earth.parseKml(kml);
16  earth.getFeatures().appendChild(kmlObject);
```

**Figure 37. Code to generate a proportional symbol map**

The API is implemented on the Thematic Mapping website as a thematic engine that is available on the <http://thematicmapping.org/engine/> website. With the Thematic Mapping Engine web interface it is extremely easy to produce a thematic map. The following thematic maps are available (Figure 38 shows the interface of the web thematic mapping engine and the result in Google Earth in Figure 39): choropleth, prism, bar and proportional symbol maps. The first option is the statistical information, the selection of the attribute and the year. This is based on data that is available on the server, thus the user cannot upload their own data. The next selection is the type of map. With each type some specific options are available:

1. Prism and Bar

The maximum size and Bar radius.

2. Symbol style

The user can decide if an image, regular polygon or 3D objects; with this the shape must be chosen if the regular polygon option is selected. The maximum size for all the types must be specified.

3. Choropleth

The colour range must be selected; the user must provide the start colour and end colour.

Classification methods are available for all map types. The following classification methods are available: unclassed, equal interval and time slider. Lastly, the display options: show the title and source, show a colour legend, show values, and show names. The interface is easy to use, but the drawbacks are that there are limited classification methods available, and, similar to the API, there is no option to select the number of classes.

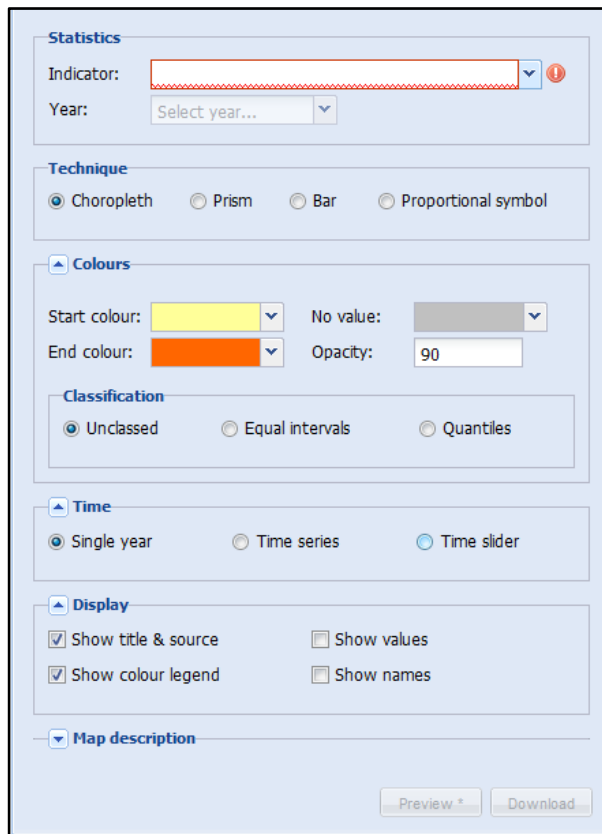


Figure 38. Web interface implementing the Thematic Mapping API



Figure 39. A bar chart showing the population per country produced with the Thematic API, displayed in Google Earth

### 6.2.3.3. *indiemapper*

indiemapper was launched in 2010 by Axis Maps as a pay per use service, but in the beginning of 2012 Axis Maps made it available free of charge (<http://indiemapper.com/>). indiemapper is an easy and flexible thematic mapping online application that aids in producing static maps through traditional cartographic design. The user interface of indiemapper is well designed and user friendly, allowing non-experts to effortlessly access advanced GIS functionalities provided by the application.

indiemapper allows the user to upload data or make use of the numerous data files available in the data library (see Figure 40). The following example was produced with economic data provided by the World Bank.

indiemapper can produce the following thematic map types: choropleth, proportional symbol, dot density and cartogram maps. Figure 41 shows the interface for selecting the thematic map type, attribute to be mapped, and whether the data should be standardised. The options are the same for all polygon vector types, but with point data only proportional symbol maps can be produced.

Figure 42 depicts the produced map and styling options for a choropleth map. A choropleth map can be either classed or unclassed. For both options, ColorBrewer colour schemes are available for the user to choose from, ensuring that the map is aesthetically pleasing. For a classed map, equal interval, quantile, optimal breaks and manual classification methods are available, with classes between 2 and 8. Figure 43 depicts the same interface but for proportional symbol maps, the difference is that the user can specify the size of the symbols displayed on the produced map.

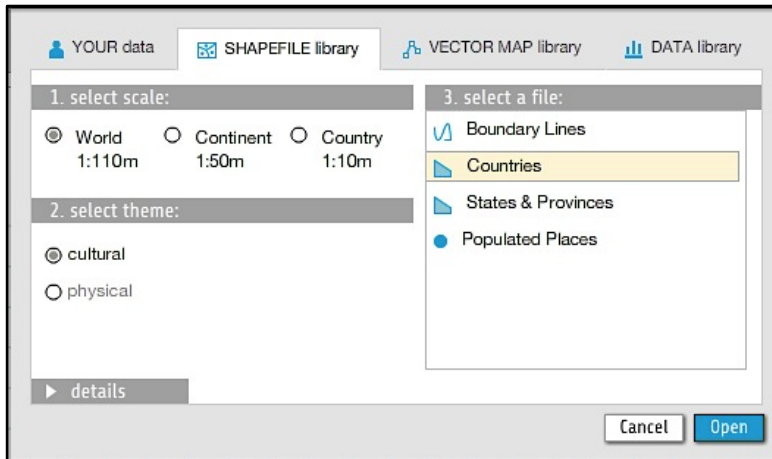


Figure 40. indiemapper interface to select spatial data

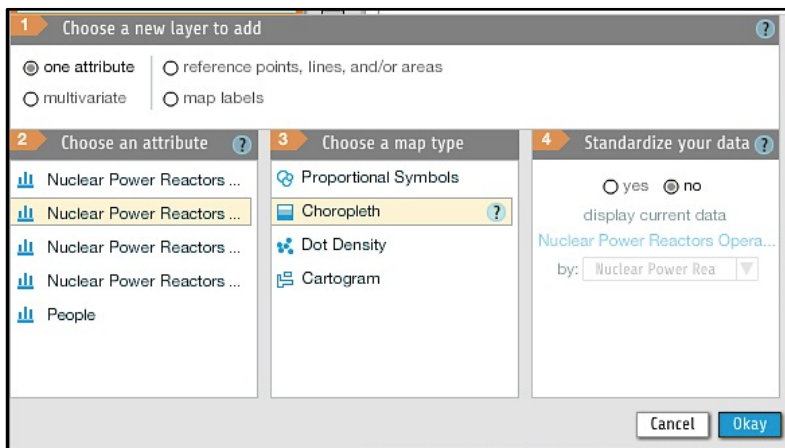


Figure 41. Interface for selecting the thematic mapping type and attribute to be mapped

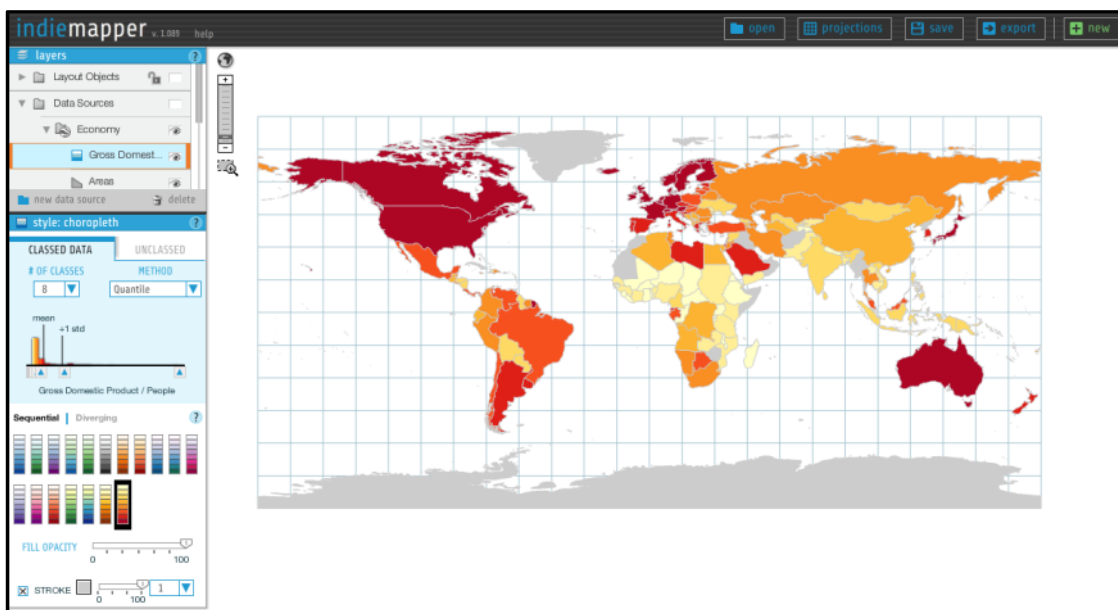
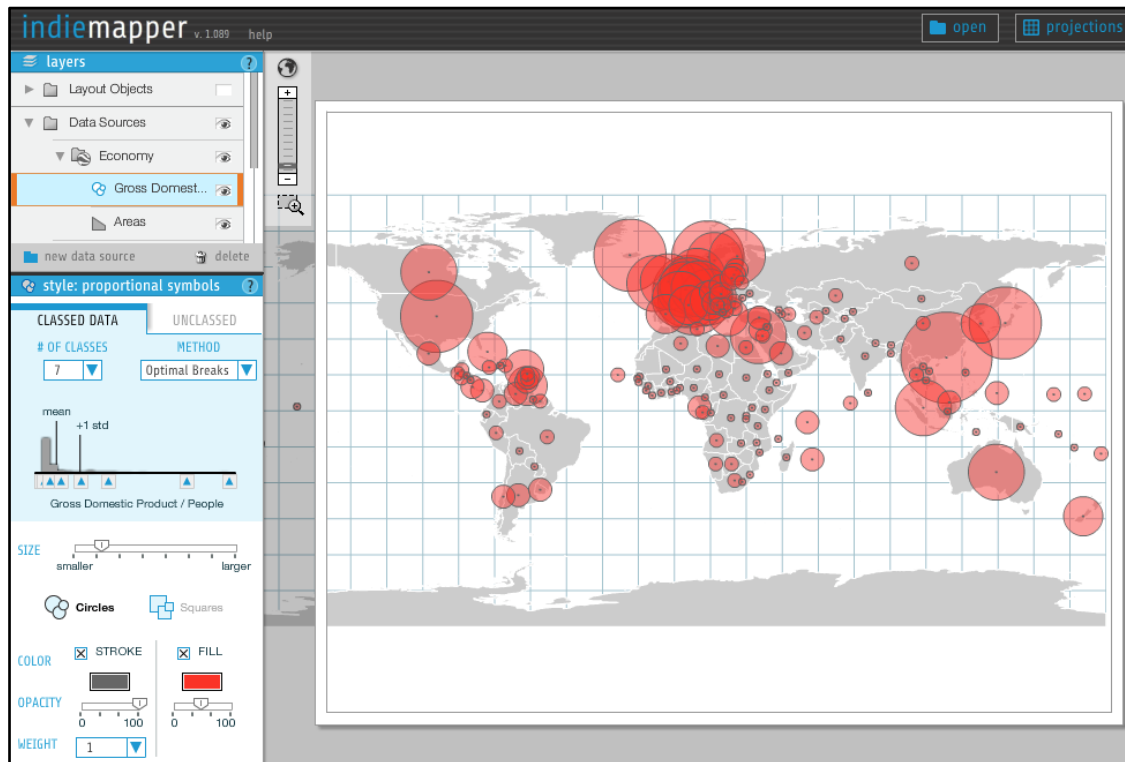


Figure 42. indiemapper with styling options and produced choropleth map





**Figure 43. indiemapper with styling options and produced proportional symbol map**

#### 6.2.3.4. GeoCommons

GeoCommons is an open repository of spatial data and maps created by a community of geographic users around the world. GeoCommons allows users to upload, analyse and share spatial data as well as maps produced. The interface is created in such a way as to be easy and simple to use for users of all expertise levels.

Figure 44 shows the styling options, legend and produced map in the GeoCommons interface. The user has the option to select the attribute to be mapped, classification method, colour schema and number of classes. GeoCommons provides four classification methods: equal interval, quantile, mean-standard deviation and maximum breaks, with little histograms depicting ideal distribution of data for the method. Predefined colour schemes are available for the user to choose from. Process and inputs required for producing a proportional symbol map are similar to that of a choropleth map (refer to Figure 45), except that the user has the option to select the maximum size of the symbols. From the options available it can be clearly seen that GeoCommons is not a dedicated thematic mapping application, but rather focuses on sharing of data and maps.

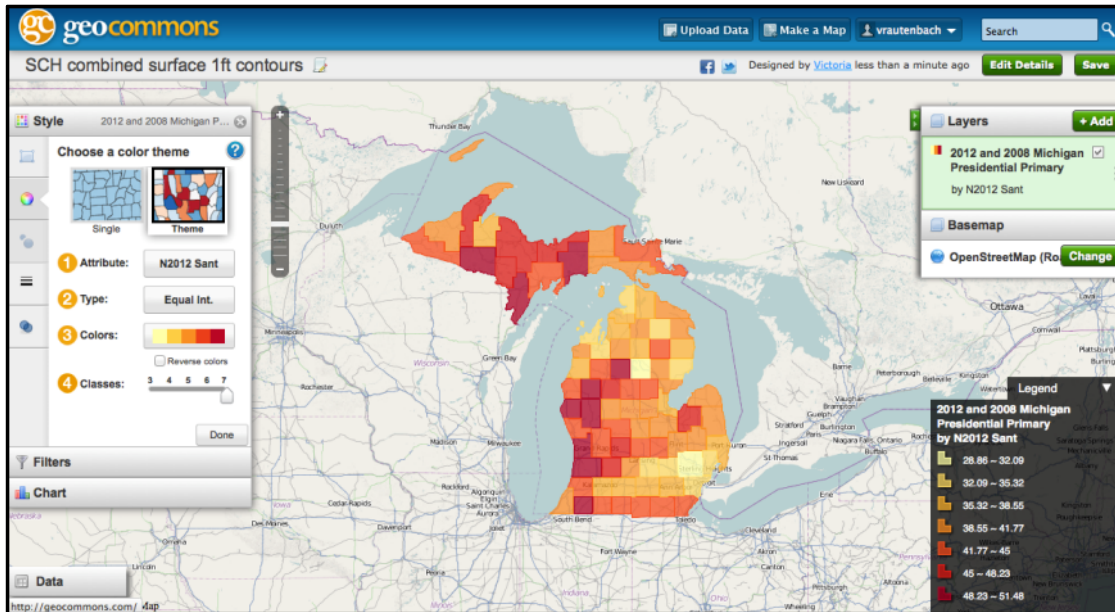


Figure 44. Interface for styling and the produced choropleth thematic map

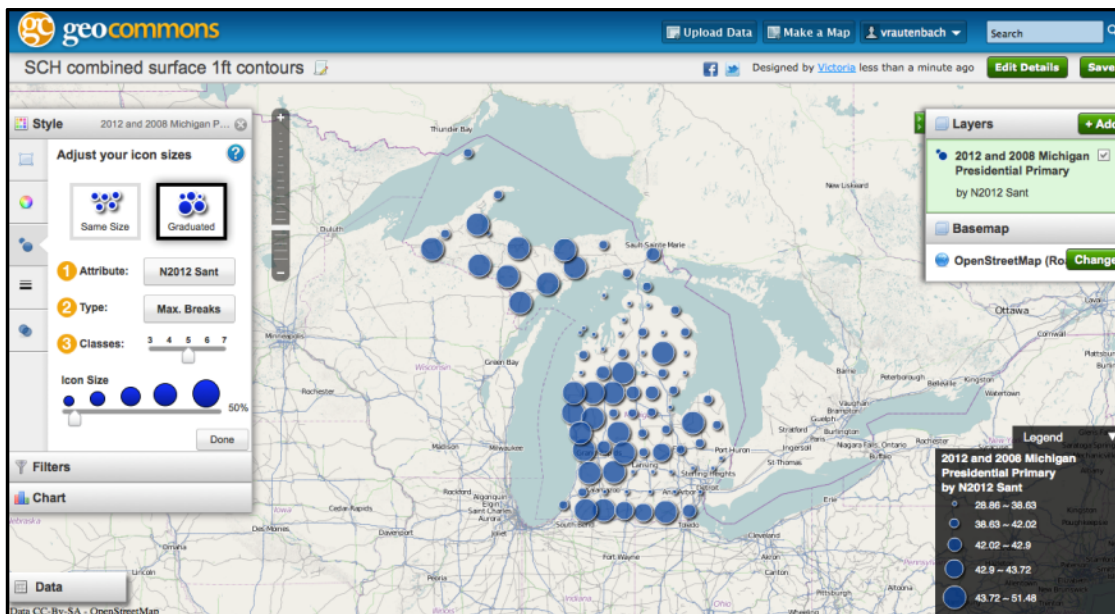


Figure 45. Interface for styling and the produced proportional symbol thematic map

### 6.2.3.5. ArcGIS JavaScript API

ArcGIS JavaScript is a freely available JavaScript API from ESRI that enables users to create high performance, easy-to-use web mapping applications. The API allows novice users to easily embed maps inside their web pages. The ESRI resource centre provides numerous code snippets and samples to get users of any level started with the API (ESRI 2010).

The ArcGIS JavaScript API defines a group of classes named Renderers. A renderer defines a set of symbols that will be used for graphics in a layer. A renderer can be used to symbolize features with different colours or sizes based on a particular attribute. The API defines three renderers: simple, class break, and unique value renderer.

The simple renderer uses the same symbology for every graphic in the layer. This renderer lives up to its name as the user specifies the symbol that must be used and applies it to the layer. This renderer cannot be used to produce a thematic map.

The class breaks renderer symbolizes each feature within a layer according to the value of some attribute associated with the layer (refer to Figure 46 and 47). Features with similar values would thus get the same symbology. The numeric attribute is divided into breaks and, for each break, a specific symbology is given and the layer is then drawn accordingly. The class contains two methods:

```
addBreak(minValue, maxValue, symbol)
removeBreak(minValue, maxValue)
```

The Unique value renderer symbolizes groups of features with matching attributes (refer to Figure 48). This technique is most common with nominal or string data. All values with the same value would get the same symbology. The class contains two methods:

```
addValue(value, symbol)
removeValue(value)
```

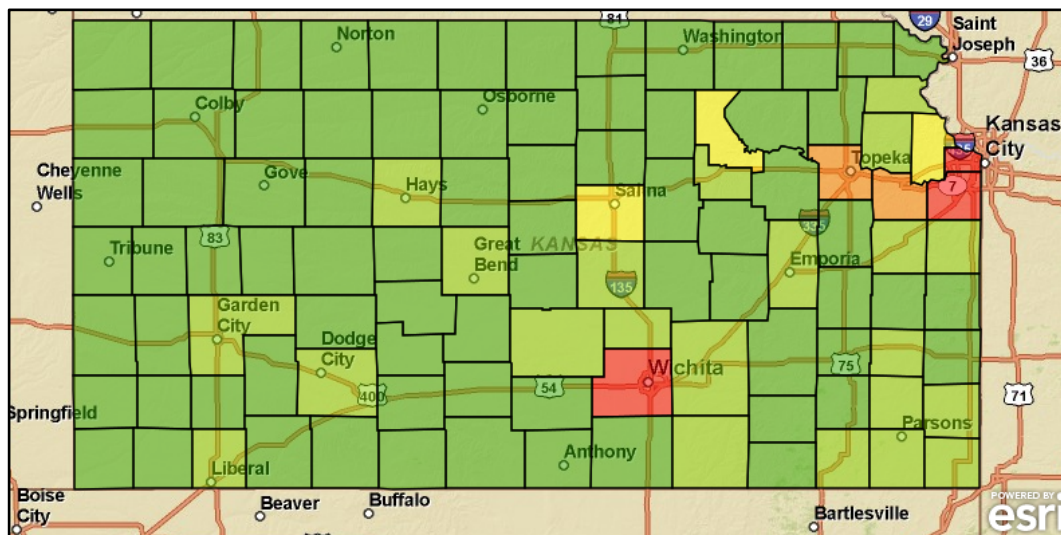


Figure 46. A class break renderer choropleth map example. Courtesy of ESRI Resource Centre

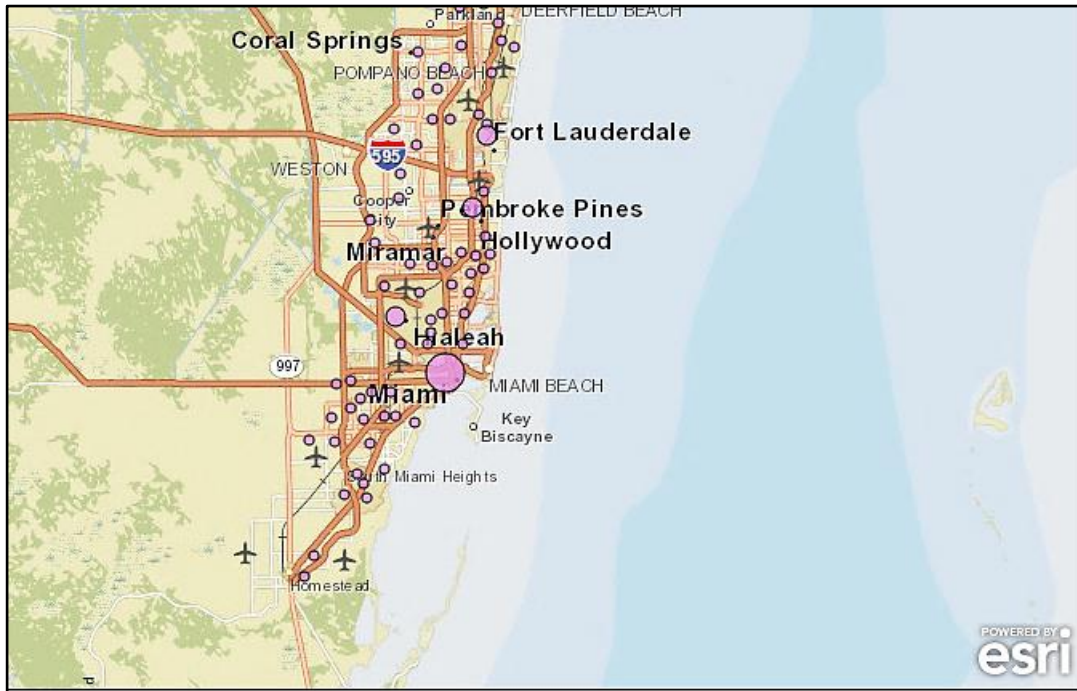


Figure 47. A class break renderer choropleth map example. Courtesy of ESRI Resource Centre

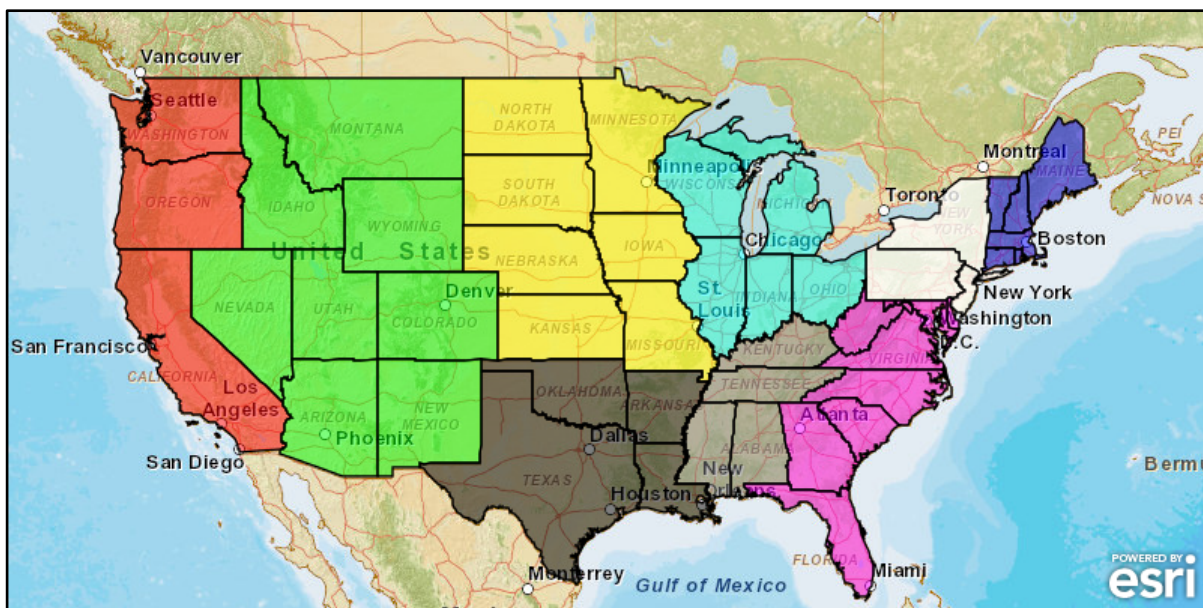
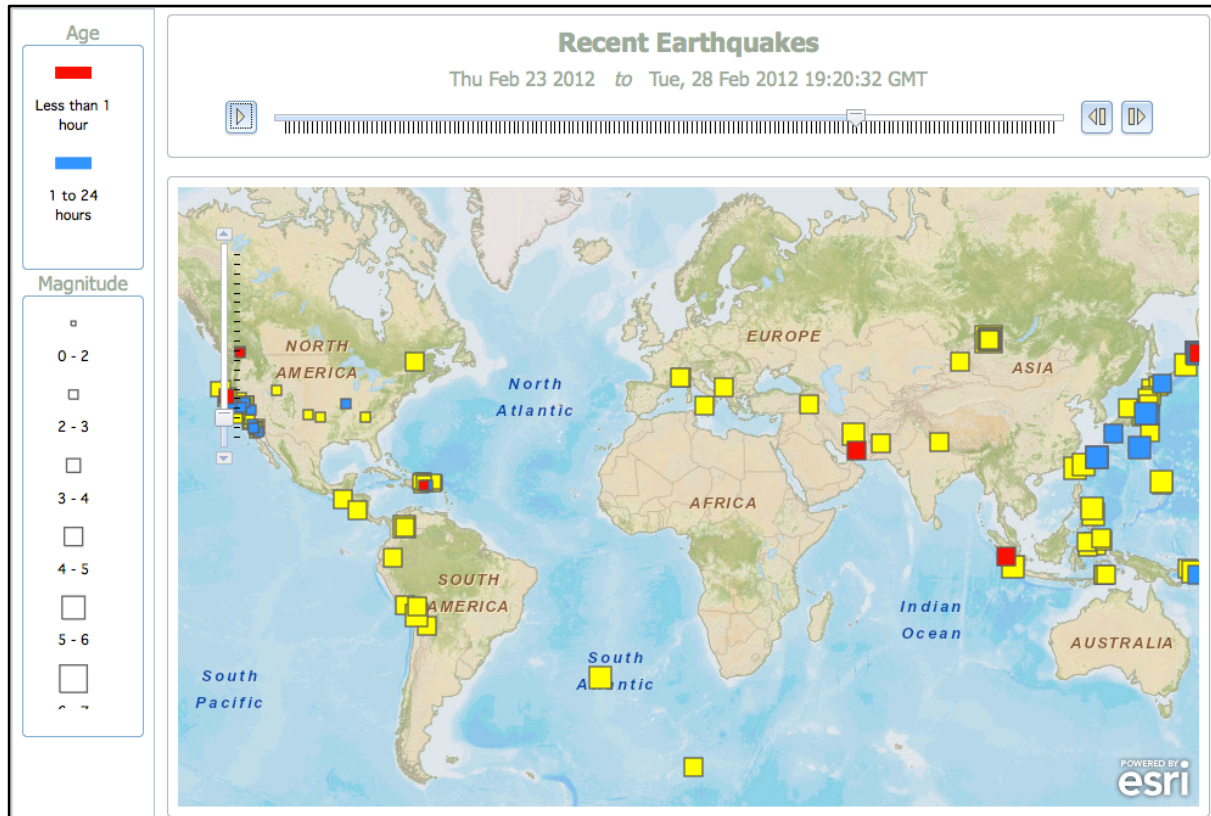


Figure 48. A unique value renderer example, courtesy of ESRI Resource Centre



**Figure 49. Animated map example, courtesy of ESRI Resource Centre**

Animated maps can also be created with the ArcGIS JavaScript API. Figure 49 depicts an example available on the ESRI resource centre web site.

#### **6.2.3.6. Google Charts API**

Google Chart API is a simple API made available by Google that enables the creation of charts from specified data and embeds it into a webpage. The API creates a PNG chart image from the data. Google Chart API is used within the Thematic mapping API (refer to Section 6.2.3.2) and GeoServer chart extension (refer to Section 6.2.3.1 and 3.5.3.3). Table 4 specifies the parameters required to create a chart with the Google Charts API.

**Table 4. Parameters needed to create a chart with Google Chart API**

Parameter	Example	Description
Base url	<a href="http://localhost">http://localhost</a>	
Cht	P3	The Chart Type, example 3D Pie Chart
Chs	24	Chart Size
Chd	T:60, 40	Chart Data
Chl	8	Slice Labels

#### **6.2.4. Results of the evaluation**

A brief summary of the evaluated implementations and standards are provided in the section. The summary focuses on the following evaluation criteria set out in Section 6.2.1: thematic cartography methods implemented, classification methods available, data standardisation methods available, and visualisation options.

Desktop applications are currently the most widely used thematic cartography implementation for the production of thematic maps. Some of the main reasons for this is the simplicity of the application, and processing power available on desktop computers. The two popular desktop applications evaluated were ArcGIS and QGIS. There are distinct differences between the interfaces for producing thematic maps:

1. QGIS requires the user to select a type of classification and to specify the maximum size of the symbols.
2. QGIS provides the user with more freedom, but requires a more experienced user than ArcMap.
3. In QGIS, the user can upload their own symbols, or a symbol can be chosen from a wide range of available symbols; however, the user cannot change the symbol colour. ArcGIS has a number of predefined shapes that can be used and the user can change the symbol colour.

##### **6.2.4.1. Thematic cartography methods implemented**

Table 5 summarises the thematic cartography methods employed in the implementations and standards; however, only choropleth, proportional symbol, dot density, diagram, and cartogram maps are shown in the table since these methods are the most popular thematic map types. Other types of thematic maps may be also be included in the application, but are not summarised in Table 5.

##### **6.2.4.2. Classification methods available**

Table 6 shows the classification methods employed by the applications. It can be seen that SLD have no specific classification methods since the user produced the rule manually and thus has to do the classification.

##### **6.2.4.3. Data standardisation methods available**

Whether or not data standardisation is available in an application is shown in Table 7. The table only displays a yes/no answer, since data standardisation is not a common function, and usually the application does not allow the user to select the specific method which is used, only the attribute on which the data standardisation should be performed.

**Table 5. Thematic cartography methods available in the evaluated implementations**

	Choropleth maps	Proportional symbol maps	Dot density map	Diagram map	Cartogram
ArcGIS	✓	✓	✓	✓	
QGIS	✓	✓	✓	✓	
OpenGeoDa	✓				✓
Standard SLD <sup>1</sup>	✓	✓		✓ <sup>2</sup>	
SLD extensions in GeoServer	✓	✓	✓	✓	
Thematic API (KML) <sup>3</sup>	✓	✓		✓	
indiemapper	✓	✓	✓		✓
GeoCommons	✓	✓			
ArcGIS JS API	✓	✓			

<sup>1</sup> The user must create an SLD style sheet manually in XML that consists of rules and filters.

<sup>2</sup> Only possible in GeoServer with the Google charts extension.

<sup>3</sup> Can only be used in virtual globes, such as Google Earth that are able to read KML files.

**Table 6. Classification methods available in the evaluated implementations**

	Unique values	Equal intervals	User-defined intervals	Quantiles	Natural breaks (Jenks)	Maximum breaks	Pretty breaks	Mean-standard deviation	Geometric intervals
ArcGIS	✓	✓	✓	✓	✓			✓	✓
QGIS	✓	✓		✓	✓		✓	✓	
OpenGeoDa	✓	✓		✓	✓			✓	
Standard SLD <sup>1</sup>									
SLD extensions in GeoServer <sup>1</sup>									
Thematic API (KML)	✓	✓		✓					
indiemapper		✓	✓	✓	✓				
GeoCommons		✓		✓		✓		✓	
ArcGIS JS API	✓		✓		✓				

<sup>1</sup> The user must perform the classification separately, after which the rules must be created.

**Table 7. Data standardisation available in the evaluated implementations**

	<b>Is data standardisation possible?</b>
ArcGIS	✓
QGIS	x
OpenGeoDa	✓
Standard SLD	Data standardisation must be performed separately
SLD extensions in GeoServer	Data standardisation must be performed separately
Thematic API (KML)	x
indiemapper	✓
GeoCommons	x
ArcGIS JS API	Data standardisation must be performed separately

#### **6.2.4.4. Visualisation options**

For thematic maps, the symbolisation and visualization of the map is important. Symbolisation has a number of aspects. In Table 8, a summary shows whether the application allows the user to select the colour range or upload own symbols for proportional symbol maps. The second section of the table shows what map elements are available, and whether the user can organise the elements to create a custom map layout.

#### **6.2.5. Conclusion**

From the results in Table 5, it is clear that the most implemented thematic map types are the choropleth and proportional symbol map and the decision was then made to develop the ThematicWS service specifically for the production of choropleth and proportional symbol maps.

Due to the limited number of classification methods implemented in web mapping applications (refer to Table 6), there is a need for histograms such as the one implemented in ArcGIS (refer to Figure 29) to support in the decision making of the classification method. However, the implementation of a histogram service is not within the scope of this research; this is covered by data exploration rather than a thematic web service. Data standardisation methods are only implemented in three of the evaluated implementations and standards. At present, visualisation options are inadequate in web mapping applications compared to that of desktop applications (refer to Table 8). In the web application, map images are produced with no additional map elements available; the produced maps are currently only used for online viewing.

The results show that desktop applications provide a number of additional functionalities and cartographic methods over web mapping applications and APIs. This can be contributed to the lack of spatial web services, and performance issues relating to web mapping.



**Table 8. Visualisation and symbology available in the evaluated implementations**

		ArcGIS	QGIS	OpenGeoDa	Standard SLD	SLD extensions in GeoServer	Thematic API (KML)	indiemapper	GeoCommons	ArcGIS JS API
<b>Symbolisation functions available</b>	Can custom colours be selected?	✓	✓				✓	✓	✓	✓
	Can custom images be selected?	✓	✓							✓
<b>Map layout options available</b>	Legend	✓	✓	✓	✓ <sup>1</sup>	✓ <sup>1</sup>	✓	✓	✓	✓ <sup>2</sup>
	North arrow	✓	✓							✓ <sup>2</sup>
	Scale bar	✓	✓	✓						✓ <sup>2</sup>
	Title	✓	✓				✓			✓ <sup>2</sup>
	Can the map layout be customised?	✓	✓							✓ <sup>2</sup>
	Can the map be exported as an image?	✓	✓	✓	✓	✓	✓	✓	✓ <sup>3</sup>	✓
Can the map be exported, including all map elements?	✓	✓								

<sup>1</sup> GetLegend method is available in the WMS SLD service

<sup>1</sup> Can be generated by the user

<sup>2</sup> Map can be saved and shared with the GeoCommons community

### 6.3. Model of thematic cartography process

A thematic cartography process was modelled using the results from the evaluation in Section 6.2, theoretical thematic cartography in Section 2.7, and the map design cycle in Section 2.3.

In Section 6.2 it was decided that the choropleth and proportional symbol maps will be implemented in ThematicWS. This decision was based on the results obtained in Table 5 where these thematic map types were shown to be the most implemented. Section 6.2 briefly described the process that needs to be followed and the required parameters to produce a choropleth and proportional symbol map in each of the evaluated packages. In most cases, the user would set a number of parameters such as the classification method and symbology of the maps, which can be done in any order on the user interface, and the application would then execute the process that is unknown to the user. This was the case with applications such as ArcGIS, QuantumGIS, OpenGeoDa, Thematic Engine,

indiemapper, and GeoCommons. However, in APIs such as Thematic API and ArcGIS JS API, there are a number of methods available to the user that perform different elements of the thematic cartography process that the user can choose and call in any order. Figure 36 and 37 show how this is done using the Thematic API. These examples show that first the thematic cartography process must be decided on, followed by the classification method and then symbology.

Chapter 2 deals with thematic cartography, specifically, Section 2.7 described the thematic cartography process for producing choropleth and proportional symbol maps as found in cartography books. This provides more background to the options and parameters required in the evaluated applications, and the process that needs to be followed is clearer with the theory. When combining the theory and evaluation an outline of the model can be developed (refer to Figure 50).



**Figure 50. Outline of the thematic cartography model**

This model was refined and more detail added, such as the decisions that the user or system need to make. The resulting model is shown in Figure 51, which depicts the thematic cartography process for a choropleth and proportional symbol map. A flowchart was selected to depict the model since a number of choices are involved and a flowchart shows the different choices and the implications, and a flowchart can be effectively used when programming the different components involved in the system.

When examining the model, it can be seen that the processes required for the production of choropleth and proportional symbol maps are similar, only with minor differences. One of these differences is that the selected dataset for a choropleth map requires standardisation; however, this is optional for the proportional symbol map. The main difference between the two processes is in step 7 and 8: during the symbolisation process, colour is used for a choropleth map and symbols for a proportional symbol map.

The model can be divided into 12 distinct steps. As mentioned, some of the steps are the same for both types, but can differ as well. The following steps were identified from the model (see Figure 51):

*Step 1: Determine the goal of the thematic map.*

The user needs to determine the intended purpose and message the map should convey, as well as the audience. These decisions need to be considered in all the following steps and might influence some of the decisions made by the user, such as classification method or type of thematic map.

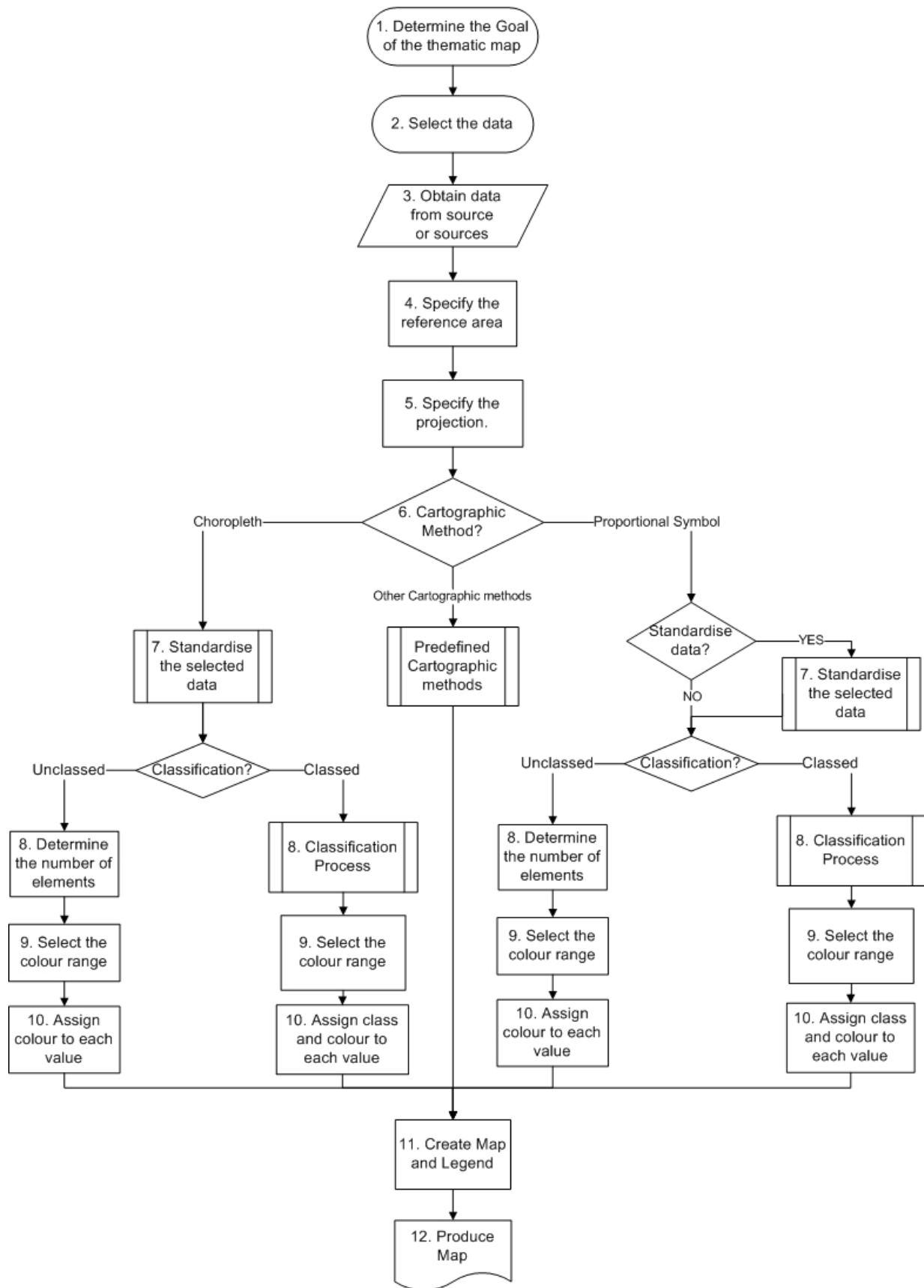


Figure 51. Flow diagram depicting the thematic cartography process

*Step 2: Select the dataset.*

In this step the user needs to select the appropriate dataset. The metadata of the dataset needs to be examined in order to determine if the dataset is appropriate. The metadata of the dataset can be examined within most geoportal implementations.

*Step 3: Obtain dataset from source.*

The dataset needs to be acquired or access to it needs to be arranged. This differs from supplier to supplier.

*Step 4: Specify the reference area.*

This is required when the dataset is larger than the area of interest. In these cases, the exact area needs to be specified as to perform the thematic cartography process only on these parts.

*Step 5: Specify the projection of the map.*

If no projection is specified, the user needs to determine the most optimal projection and transform the dataset into that projection, otherwise the user needs to ensure that the specified projection is correct.

*Step 6: Decide on the thematic cartographic method.*

As with the evaluated implementations and APIs, the first decision that needs to be made is which thematic map method is suitable. This decision relates to the goal determined in step 1; the most appropriate method should be chosen so as to achieve the goal of the map.

*Step 7: Standardise the dataset, if required.*

In Section 2.5, standardisation of data for thematic mapping is discussed in more detail. To summarise, standardisation is required in order to obtain a more meaningful map. Standardisation is a requirement for choropleth maps; however, if the data is not standardised the map will still be produced but might contain some errors. For proportional symbol maps, data standardisation is optional, which depends on whether the data requires standardisation. The user or expert system will need make this decision.

*Step 8: Classify the dataset, if the user selected this option.*

For both methods, the user needs to determine whether the map will be classed or unclassed. This decision should be made with the goal of the map in mind as well as the number of values within the dataset. With unclassed thematic maps each unique value is assigned a unique symbology (determined in step 9 and 10), but with classed maps the classification method and number of classes must be specified. The choice of classification method and number of classes are mainly not mutually exclusive. These two decisions influence each other and should thus be considered together.

*Step 9 and 10: Symbolisation process.*

The symbology of the map is what makes the map a choropleth or proportional symbol thematic map. As mentioned in Step 8, if an unclassed map is selected, each unique value will be assigned a unique symbology whether that is for a choropleth or proportional symbol map. However, for classed maps, each class will have a unique symbology. The most significant difference between a choropleth and proportional symbol map lies with the symbology of these maps. For choropleth maps, the area of the polygon will be symbolized according to a class, thus only a colour of this is required. But for proportional symbol maps, a symbol relative to the value will be placed near the centre of a polygon, the symbols having a colour and maximum size. The symbol used in proportional symbol maps can also be defined. Commonly the symbols are well-known symbols or can be user uploaded. Depending on the platform on which the thematic maps are created, different technologies are used to create the styling.

*Step 11 and 12: Produce map and additional information, e.g. legend and layout.*

In this step the map is created with the specified styling determined in step 9 and 10, and other map elements such as a legend is generated and added to the map.

It is important to understand the steps involved in producing the thematic maps before attempting to orchestrate web services to automate the process. Figure 51 depicts the model of the thematic cartography process. This model can now be used to determine the required web services for the development of the orchestrated service, ThematicWS. Web services can be orchestrated to perform the entire process depicted in Figure 51. However, ThematicWS will only perform steps 3, and 7 through 12 and in our implementation the cartographer still has to perform steps 1, 2, 4 to 6, and after the map has been produced, the cartographer needs to validate that the map is correct and conveys the intended message as specified in the goal (refer to Step 1).

## **6.4. Conclusion**

The model is unique in that it divides the thematic cartography process into easy to follow and implement steps. Theoretical thematic cartography is explained by many authors as separate components (map design, symbology, classification, etc.) forming the thematic process, but has rarely been combined in this fashion to demonstrate a sequence of steps to produce thematic maps.

The thematic cartographic model developed provides a framework for the orchestrated thematic web service, ThematicWS. The model focuses on the production of choropleth and proportional symbol maps, but can be easily extended to include isarithmic, dasymetric, and dot density maps. This is possible because these types of thematic maps are a variation of the choropleth and proportional symbol maps. The model is limited in that it does not take into consideration best practices. If the model is combined with best practices, it could lead to an implementation framework for the implementation of an expert system for the production of choropleth and proportional symbol maps.

## Chapter 7 Design and implementation of ThematicWS

### 7.1 Introduction

The model of the thematic cartography process was established in Chapter 6, which provides a step-by-step process that can now be transformed into a service that will automate the process. In this Chapter the technical and non-technical requirements are specified and architectural design is determined and refined in the detailed design.

### 7.2 Functional requirements

Functional requirements capture the intended behaviour of ThematicWS service (Tsui & Karam, 2007). The behaviour can be expressed as services, tasks or functions that the ThematicWS needs to perform. The requirements are divided into two categories: user requirements and system requirements. User requirements refer to functionalities that the user requires ThematicWS to perform, and system requirements the internal functionalities required to achieve the goal and objectives of ThematicWS.

#### 7.2.1. User requirements

##### 7.2.1.1. Produce a thematic map

The ThematicWS service shall be able to produce the following types of thematic maps:

1. Unclassed choropleth map
2. Classed choropleth map
3. Unclassed proportional symbol map
4. Classed proportional symbol maps

In Chapter 6, applications for the production of thematic maps were investigated. In Section 6.2.4.1, a summary is provided on the number of implementations of different thematic cartographic methods. The results showed that the choropleth and proportional symbol maps are commonly implemented.

##### 7.2.1.2. User options available

For each of the following thematic maps, the user can select the following options:

1. Unclassed choropleth map
  - 1.1 The attribute according to which the maps must be produced.
2. Classed choropleth map
  - 2.1 The attribute according to which the maps must be produced.
  - 2.2 The number of classes the values should be divided into.
  - 2.3 Classification method, such as equal interval or quantile.
3. Unclassed proportional symbol map
  - 3.1 The attribute according to which the maps must be produced.

4. Classed proportional symbol maps
  - 4.1 The attribute according to which the maps must be produced.
  - 4.2 Whether the data should be standardised or not.
  - 4.3 The number of classes the values should be divided into.
  - 4.4 Classification method, such as equal interval or quantile.

#### **7.2.1.3. Symbolization options for proportional symbol maps**

For each of the two types of proportional symbol maps, the following symbolization options shall be available to the user:

1. Unclassed proportional symbol map
  - 1.1 The user can specify the symbol that should be used; a well-known symbol type, such as circle or square.
  - 1.2 The user can specify the colour of the symbol.
2. Classed proportional symbol map
  - 7.5. The user can specify the symbol that should be used; a well-known symbol type, such as circle or square.
  - 7.6. The user can specify the colour of the symbol.

#### **7.2.1.4. Symbolization options for choropleth maps**

For choropleth maps, there will be no symbolization options available. Predefined colour schema, from ColorBrewer shall be used to symbolize the map.

Section 7.2.1.2 to 7.2.1.4 provides an overview of the parameters required to produce choropleth and proportional symbol maps. These parameters and the associated process are described in Chapter 2 and Section 6.3.

#### **7.2.1.5. Service must be invoked using HTTP GET/POST**

The user can invoke the service using a HTTP GET/POST request.

### **7.2.2. System requirements**

#### **7.2.2.1 Retrieve attribute data from a WFS service**

The ThematicWS service shall be able to retrieve attribute data using a WFS service.

#### **7.2.2.2 Perform statistical processing**

The ThematicWS service shall be able to compute the necessary statistics required to standardise and classify the attribute data.

#### **7.2.2.3 Generate a custom SLD style sheet**

Using the information received from the statistical processing such as the class breaks, a custom style sheet shall be generated to satisfy the user's specification.

#### **7.2.2.4 Generate a thematic map image using the custom style sheets**

The ThematicWS service shall be able to produce a thematic map image, such as jpeg, of the map generated using the attribute data and the generated SLD style sheet.

### **7.3 Non-functional requirements**

The non-functional requirements described the manner in which the service shall behave (Tsui & Karam, 2007).

#### **7.3.1 Open Source**

The ThematicWS service shall be developed completely using open source technologies and under an open source license, such as GNU General Public License or GPL version 2 or later. It should also be ensured that the service developed and license selected conforms to the licenses employed by technologies used.

#### **7.3.2 Performance requirements**

ThematicWS should be developed in such a way as to be optimal for executing on a server machine. The performance of the service will depend on numerous factors, such as the other service it will rely on and the size of the dataset, thus making performance a difficult issue to address. The code developed for the ThematicWS should be optimized where possible.

#### **7.3.3 ThematicWS shall consist of standards OGC web services**

ThematicWS shall be developed using implementations of standard OGC web services, such as WMS, WFS, and WPS services. Existing standard OGC service implementations of WMS and WFS from GeoServer should be used.

#### **7.3.4 Interoperability and modifiability requirements**

As specified in the objective, ThematicWS is an orchestrated service, which makes interoperability a very important consideration. ThematicWS relies on syntactical interoperability of OGC web services. The service shall be modular and allow for the interchangeability of web services, thus a standard OGC web service can be exchanged for another standards OGC web service. For example, a WMS service provided by GeoServer may be exchanged with a WMS service from deegree.

This new orchestrated service, ThematicWS, should be designed so that it can easily be integrated in a geoportal of an SDI.

#### **7.3.5 ThematicWS shall be created through orchestration of services**

The standard OGC web services shall be orchestrated to develop the new service, ThematicWS. The orchestration process may be done using methods, such as workflows script or workflow modelling.



## 7.4 Architectural design

A high-level overview of the ThematicWS service is presented. The architectural design presents the first attempt to transform the functional requirements (refer to Section 7.2) and non-functional requirements (refer to Section 7.3) to the design of the service. The high-level architectural design of ThematicWS will provide input for the detailed design.

Two architecture types were considered for the implementation of ThematicWS: distributed component architecture and service oriented architecture (SOA). Sommerville (2010) defines distributed component architecture as a software system developed by composing independent, deployable components that execute on different processors used when different systems or databases need to be combined into a single composed system. A component is a unit of composition that conforms to a standard component model or interface and can be independently deployed and composed without modification according to a composition standard (Councill & Heineman, 2001; Szyperski, 2002). Components are similar to services. If the system requires a service, it can call on a component to provide the service without any knowledge about the component. The distributed component system relies on middleware that manages the component interactions, resolves parameter conflicts, and provides a common set of services that can be consumed by other components (Sommerville, 2010).

SOA is an architecture software concept which is used to develop distributed systems of which the components are independent services, executed on geographically distributed computers (Sommerville, 2010). SOA defines the use of services to implement specific business processes. Resources in an SOA environment are exposed as independent services that can be accessed through a standardised interface. Service based applications can be constructed by composing services from different service providers which can be achieved through using standard programming languages or workflow modelling.

The distributed component architecture and SOA are similar in that both promote loosely coupled and a highly interoperable software architecture that enable efficient, reusable, and error-free software development, thus there is no clear boundary between these architecture models. SOA can be considered an evolution on the distributed component architecture in that it implements many of the distributed component architecture characteristics; however, SOA focuses on service-based applications.

ThematicWS will be implemented using both SOA and distributed component architecture principals and methodologies. ThematicWS shall be developed using new services wrapped in OGC WPS services and existing standard OGC web service implementations. The custom WPS services should be reusable. Sommerville (2010) defines the following stages in the process of service composition:

1. Formulate outline workflow

The initial design of the service is produced. The requirements (refer to Section 7.2 and 7.3) are used to create an abstract design of the orchestrated service.

2. Discover services  
During this stage, existing services are looked up and discovered using various methods such as registries.
3. Select possible services  
The previously discovered web services are evaluated according to a set of user-defined criteria. This evaluation is used to determine whether the web service can be used in ThematicWS.
4. Refine workflow  
Using the information gained on the selected web services, the workflow is refined.
5. Create workflow program  
The abstract workflow designed is transformed into an executable program, and service interface is defined.
6. Test completed service or application  
Test the complete ThematicWS service.

In this chapter, stages 1 through 4, stage 5 in Chapter 9 and stages 6 in Chapter 10 are discussed.

#### 7.4.1. Formulate outline workflow

In this stage, the initial abstract design of the orchestrated web service is developed. The input for this stage are the functional requirements (refer to Section 7.2), non-functional requirements (refer to Section 7.3), and the thematic cartography model (refer to Section 6.3). Figure 52 depicts the simplified version of the thematic cartography process that shall be automated in ThematicWS for the production of choropleth and proportional symbol maps. The figure does not provide detail but only highlights the main components. The thematic cartography process (refer to Figure 51) can be divided into the following steps, as shown in Table 9. The corresponding web service standard is also indicated.



**Figure 52. Simplified step of ThematicWS**

ThematicWS will perform steps 3 and 7 through 12. In this implementation the cartographer still has to perform steps 1, 2, 4 to 6, and after the map has been produced the cartographer needs to validate that the map is correct and conveys the intended message. An initial workflow is depicted in Section 8.5, Figure 62.

**Table 9. Step in thematic cartography process with corresponding OGC web service**

Thematic cartography process	Standard
Step 1: Determine the goal of the thematic map.	Not applicable
Step 2: Select the dataset.	Not applicable
Step 3: Obtain data from source.	WFS gets the data
Step 4: Specify the reference area.	Not applicable
Step 5: Specify the projection of the map.	Not applicable
Step 6: Decide on the thematic cartographic method.	Not applicable
Step 7: Standardise the dataset, if required.	<b>A standard is required</b>
Step 8: Classify the dataset, if the user selected this option.	<b>A standard is required</b>
Step 9 and 10: Symbolisation process.	SLD allows symbolisation but has limitations for dynamic thematic mapping.
Step 11 and 12: Produce map and additional information, e.g. legend and layout.	WMS renders a thematic map

#### 7.4.2. Discover and selecting possible services

The aim of this stage is to search and discover web services that could possibly be used in the orchestrated service, ThematicWS. However, for this project this stage is not necessary, since the non-functional requirements specify that WMS and WFS services implemented in GeoServer shall be used. For step 7 to 10 in Table 9, the custom functionalities needs be wrapped in WPS services. Possible WPS frameworks were identified and 52° North and ZOO project were selected. Both frameworks are open source and comply with the WPS 1.0 standard.

52° North and ZOO project's orchestration capabilities are evaluated in Chapter 8, but in this stage the WPS implementation is examined. For this evaluation, the results obtained by Garnett and Fenoy (2011) in the WPS shootout are used. They determined that both the 52° North and ZOO project framework comply with the OGC WPS standard, and interoperability of the framework was found suitable. Due to ease of installation and documentation available (refer to Table 11) of the frameworks, the 52° North WPS framework was selected.

#### 7.4.3. Refine workflow

In this stage the initial workflow defined is refined so that ThematicWS can be implemented from the information in this section. A use case diagram can be produced from the functional requirements determined in Section 7.2. The resulting use case diagram is shown in Figure 53, which illustrates the functionalities that ThematicWS exposes to the client.

The sequence diagram (refer to Figure 54) depicts the high-level workflow of ThematicWS. However, it is currently not possible to specify a detailed workflow for each of the use cases specified in Figure 53. The reason is that two of the services, *WPS for Statistical processing* and *WPS for SLD style sheet preparation* (refer to Figure 54), first need to be discussed and designed in more detail (see Section 7.5). Thereafter the sequence diagram can be refined for each use case specified.

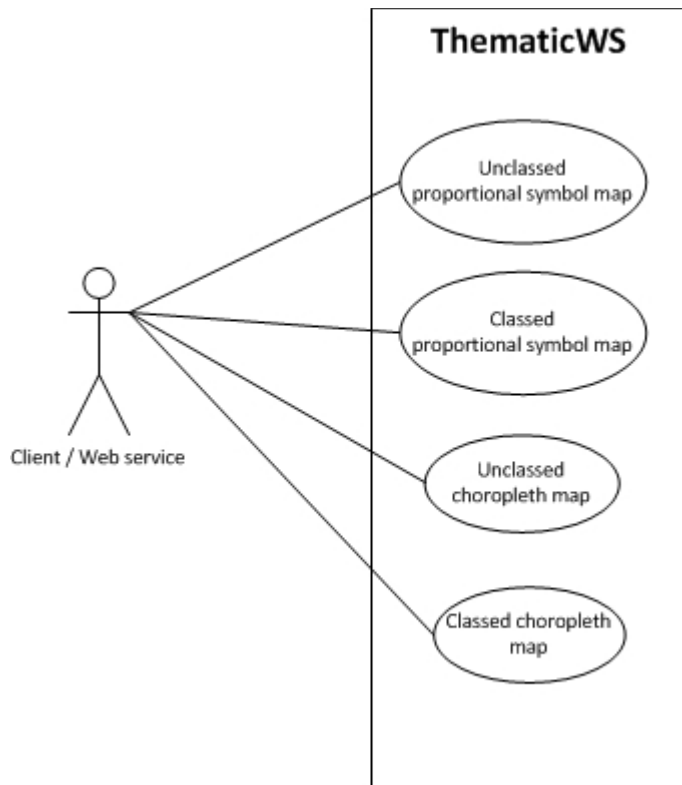


Figure 53. Use case diagram depicting the functions of the ThematicWS service

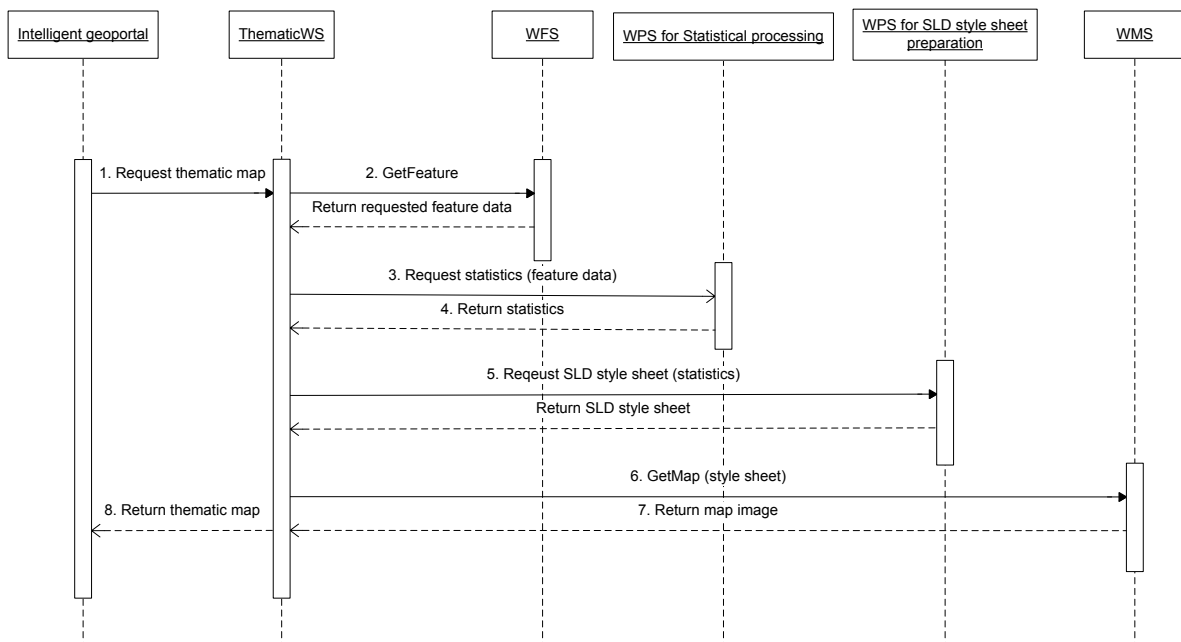


Figure 54. Sequence diagram illustrating the orchestration of web services in ThematicWS

## 7.5 Detailed design

The final design of ThematicWS is discussed in this section. This design will be used as a blueprint to implement the service.

### 7.5.1. GeoServer WMS and WFS implementation

The GeoServer implementation of the OGC WMS and WFS standards should be used in ThematicWS. OGC classifies registered products as implemented or compliant. An implemented product is defined by OGC as when the developers have made an attempt to follow its instructions regarding interface or schema syntax and behaviours as specified in the standards, and a compliant product refers to software that has passed the OGC compliance testing programs that test whether all mandatory elements as specified in the standards are implemented (<http://www.opengeospatial.org/ogc/faq/process#6>). According to the OGC implementing product details for GeoServer (<http://www.opengeospatial.org/resource/products/details/?pid=1041>), only the WMS 1.0, WMS 1.3.0, and WFS 1.1.0 are registered as implemented with no compliant products. However, the GeoServer website states that WMS 1.1.1, WFS 1.0 and web coverage service (WCS) 1.0 are compliant with the OGC standards. Even though GeoServer WMS 1.3 and WFS 1.1.0 services are only ranked as implementations by OGC, these services will be used in the development of ThematicWS.

### 7.5.2. Statistical processing service

In accordance to the non-functional requirements specified, all services used in the development of ThematicWS shall be standard OGC service implementations, thus custom services should be *wrapped* in an OGC WPS service in order to obtain a standard interface. The statistical processing service is a custom service with no alternative OGC standard available. The service will thus be implemented as a WPS service.

The functionalities of the statistical processing service are to determine a number of basic statistical information, standardising the data when required, and to classify the selected data (refer to Figure 55). The basic statistical functions implemented in this service can be extended and used for other statistical processing. The ratio standardisation method was implemented, but only performed when the user specified the need for standardisation of the dataset and, if required, parameters were provided. The equal interval, quantile and mean-standard deviation classification was selected to be implemented in ThematicWS; other classification methods can easily be added to the service. These classification methods were selected as a result of their popularity (shown in Section 6.2.4, Table 6) and ease of implementation.

The main aim of the statistical processing service is to classify the dataset into classes and generate the class limits of each. The functions were implemented manually and no libraries were used for the implementation. The reason for this is that the calculations required are fairly basic and easy to implement. The steps performed by the service are shown in Figure 56; for each method the steps are presented.

The service implements other basic statistical functions as well, such as mean, minimum, maximum, and standard deviation.. These functions will be available to the public, which aids in code reuse, in that users will not have to implement these functions manually, but can access them via the web service and quickly receive the results for large amounts of data.

The basic implementation of the statistic processing service performs the classification, but does not consider other factors to ensure that classification is optimal. To add some level of intelligence to the statistical processing service, a couple of conditions/business rules were added to ensure that the classification method performed on the data is optimal. The following conditions are created:

1. Not more than 50% of the values shall be in a single class.
2. Only one class shall be permitted to contain a single value.
3. No class shall be empty.
4. The difference between classes shall be minimal as this affects the ease of understanding the legend.

With these additional constraints the user would specify a classification method, the classification would be performed and then tested to see compliance to the pre-set conditions. If all the tests are passed, the method is proven to be optimal and result returned. However, if a test is failed, another classification method would be tested and if the method is found to be optimal, the results would be returned with an additional note stating what method classification was used. If no method satisfies all the conditions, the results of the classification method specified by the user would be used. This cannot be forced on the user, thus an additional parameter should be included in the interface of the ThematicWS service, or else the specified classification method will be used and no tests would be performed.

One of the inputs of the statistical processing WPS service is the output of the WFS service. An important consideration is in which format the data should be. The data format chosen should be an option of the output formats available in GeoServer: CSV, GML, GeoJSON, shapefile and Excel. Geography Markup Language (GML) version 3 is the default value of the GetFeature request WPS 1.1.0. GML is an XML grammar written in XML Schema for the description of application schemas as well as the transport and storage of geographic information (OGC 2007c). However, GeoJSON was selected to be the data format of the output. GeoJSON is based on JSON, which is an object notation rather than a markup language, such as GML. GeoJSON has a number of advantages above GML in data oriented services, such as: a hierarchy of objects can easily be represented as a stream, it removes redundant repetition of attribute values, and it is easier to process.

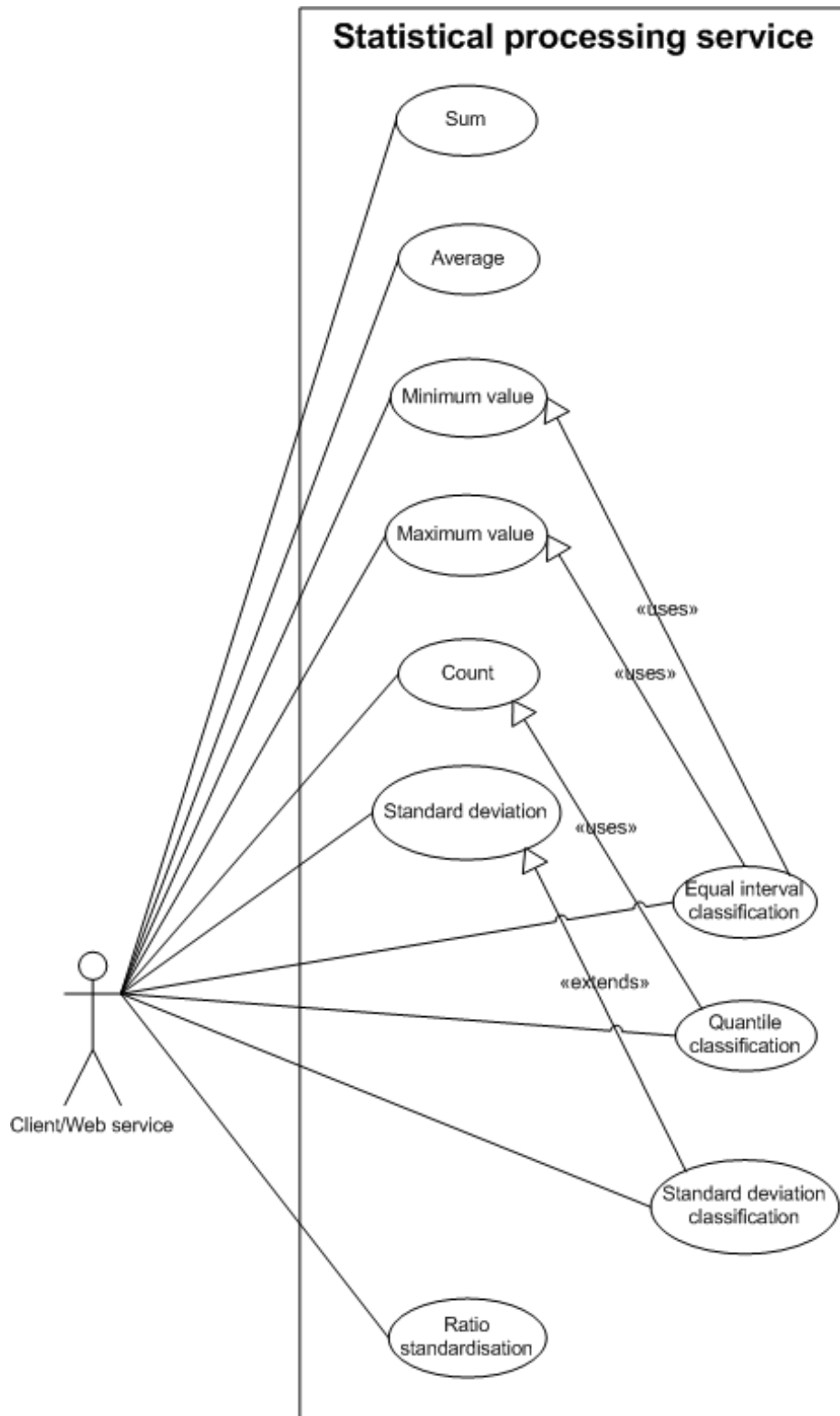


Figure 55. Use case diagram depicting the functions of statistical processing web service

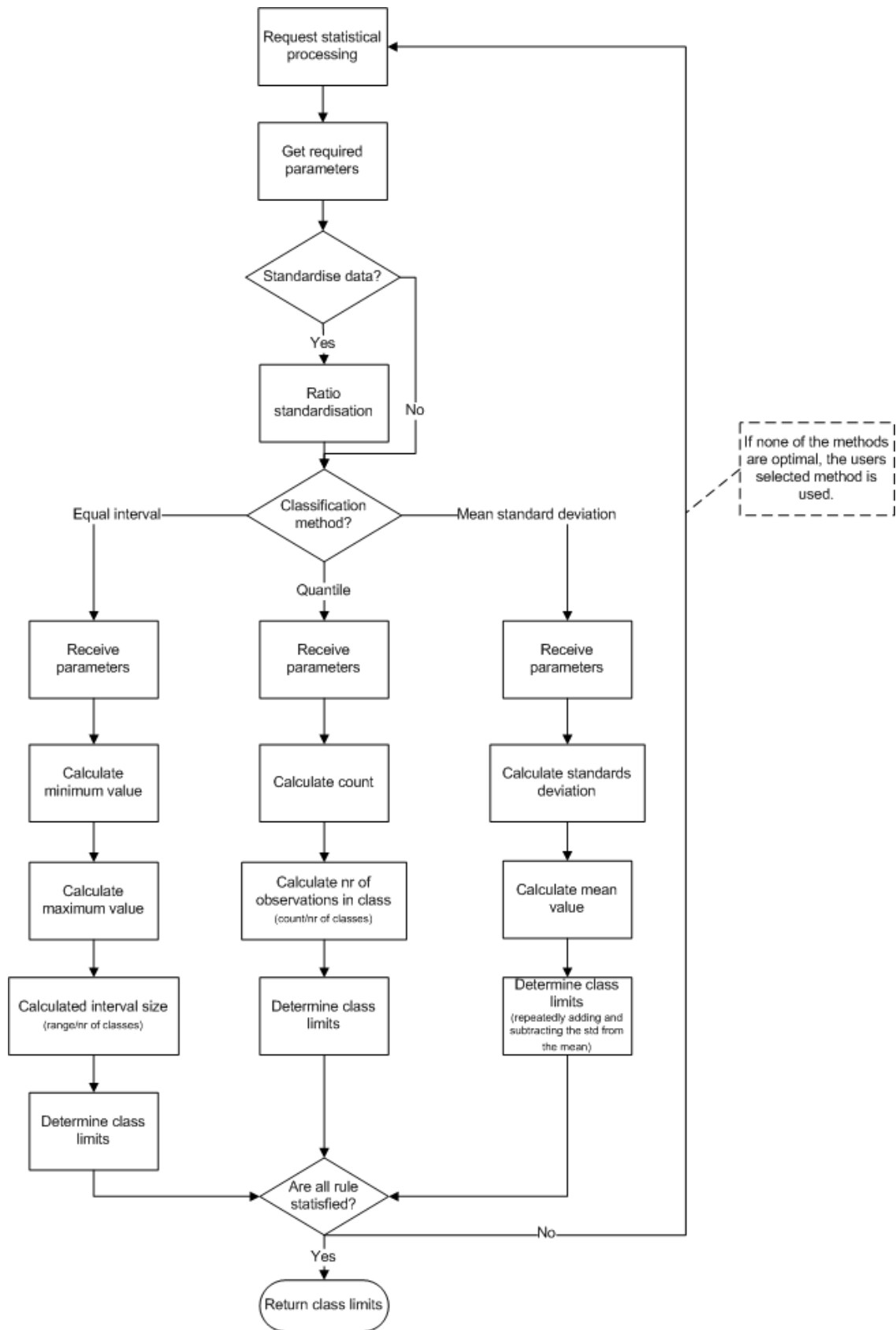
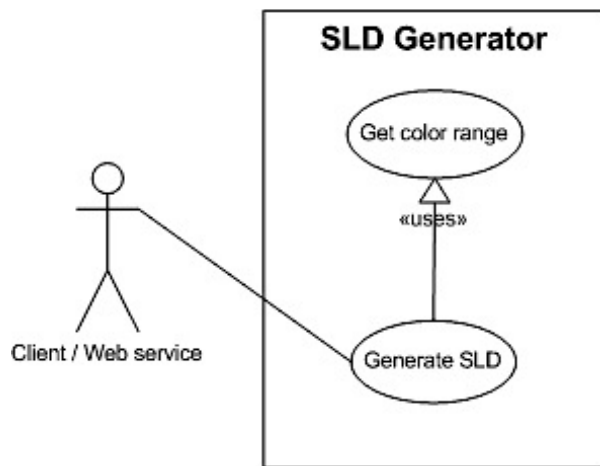


Figure 56. Process in the WPS for statistical processing



### 7.5.3. SLD style sheet preparation service

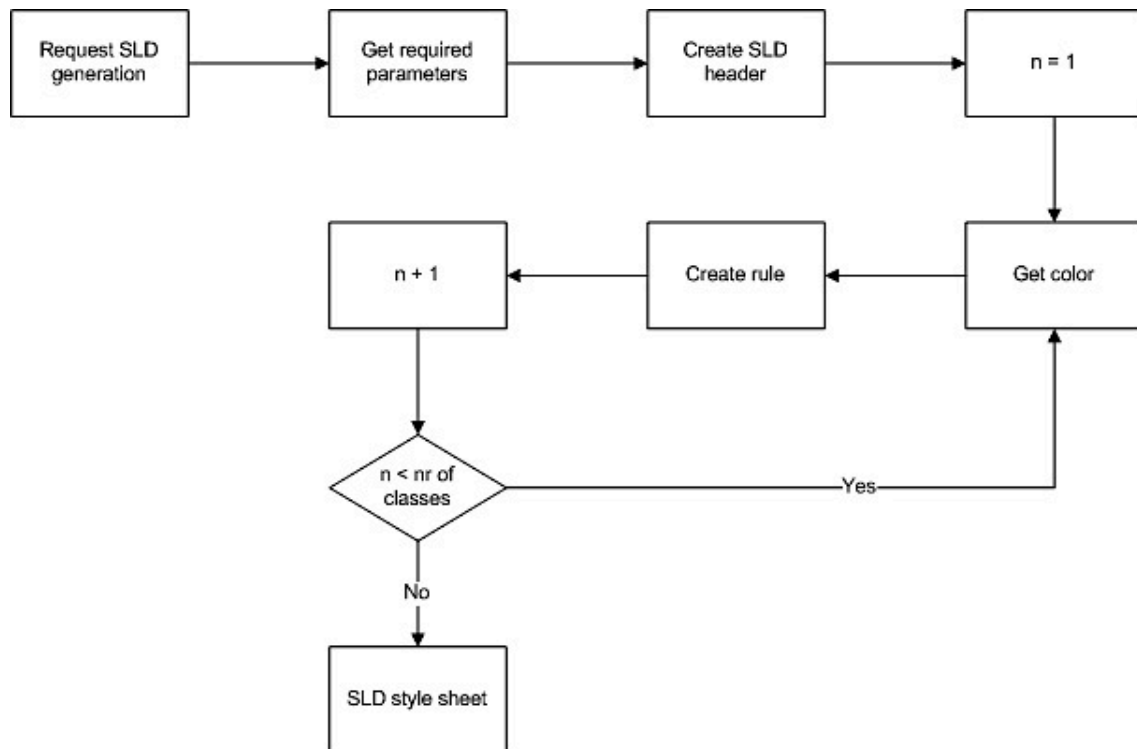
The SLD style sheet preparation service (step 4 in Figure 54) of the ThematicWS is a custom web service that should be *wrapped* in a WPS service. The service has two functions of which only one is accessible to the public (refer to Figure 57). *Generate SLD* creates a custom SLD style sheet from the options selected by the user and the classification performed on the dataset. Another function that needs to be implemented is the *get color range* function, which is private and cannot be accessed by a client. The *get color range* function determines a colour for each class within an optimal colour hue selected from the ColorBrewer website.



**Figure 57. Use case diagram depicting the functions of the SLD generator web service**

The SLD generator composes the SLD style sheets used to produce either a choropleth or proportional symbol map as requested by the user. The steps performed by the SLD generator are shown in Figure 58, which shows that the process has a loop with a counter that creates the rules in the SLD style sheet. The style sheets were manually created by composing the document that consists of a number of rules (one rule for every class). The rules can easily be created since it is a pattern that repeats with values that change, making it simple to create with a for loop function, one iteration in the loop per class defined.

In Section 3.5.3, the extensions to SLD implemented in GeoServer are explained and examples of how they can be used are provided. However, only one of the extensions is used in this implementation, the dynamic symbolizer. The dynamic symbolizer allows the user to easily create unclassed proportional symbols with only one rule. The other extensions cannot be used for this example for the following reasons: the diagram symbolizer can be used for diagram maps, but are not implemented in this example; the parameter substitution is useful when style sheets are created beforehand; and class limits are calculated at run-time and are then substituted into the style sheet.



**Figure 58. Process in the WPS for SLD generator**

#### **7.5.4. ThematicWS service**

In the previous sections, the different components of the ThematicWS service are determined as OGC WMS and WFS services combined with custom WPS web services, and explained in detail (refer to Section 7.5.1, 7.5.2, and 7.5.3). This section focuses on defining the interface of ThematicWS and the manner in which the individual services needs to be connected in order to orchestrate the service.

##### **7.5.4.1. Orchestration method**

As explained, orchestration of web services work with a central controller that executes the web services in the correct order. Two prototypes will be developed implementing different methods of orchestration, a custom orchestration engine and the use of a workflow modeller.

The first prototype will be developed using a custom orchestration engine. This is basically a java servlet that calls and transfers the data and results between the different web services. The parameters required to execute the orchestration (specified in Section 7.5.4.2) are sent through in the URL together with the GetThematic request, which is used to produce a thematic map. The orchestration engine will be simplistic in that it would call the different web services with the required parameters in the correct order, and lastly respond to the client with the thematic map image produced by the ThematicWS service.

The second prototype will be developed using the workflow modeller that is part of a WPS framework. Workflow modellers have become increasingly popular and have become part of a couple of WPS

frameworks, such as the 52° North and ZOO Project. Chapter 8 provides a detailed evaluation of the orchestration capabilities of 52° North and ZOO Project frameworks. Detail on how the orchestration was performed can be found in Chapter 8.

#### **7.5.4.2. Interface design**

In the previous sections, the different services and their parameters have been established. Figure 54 shows the sequence in which the different services will be called to create the ThematicWS service. The first OGC service that will be called is a WFS service, specifically, the WFS GetFeature request which retrieves the attribute data of the specific dataset from the server. The parameters used for the WFS GetFeature request is listed in Table 13 (in Appendix B).

WPS services will be used to standardise the custom functionalities (step 7 and 8 in Table 9). The WPS service standard provides the user with a way to publish spatial processes and expose them through a standard interface. The WPS Execute request has a number of set parameters (refer to Table 14 in Appendix B). The DataInputs parameter allows the user to send any number of input parameters in the URL to the service that will be used in the process.

Two WPS services are required to perform the statistical processing and SLD style sheet generation. The statistical processing service requires the following parameters (see Figure 56):

1. The attribute data of the layer.
2. Name of the specific attribute on which the classification must to be performed.
3. The classification method to be performed.
4. Classed or unclassified.
5. If classed, the number of classes.

The SLD generator service requires the following parameters (see Figure 58):

1. Class limits calculated by the statistical processing engine.
2. Type of thematic maps, choropleth or proportional symbol.
3. If proportional symbol, symbol type (name of well-known symbol).
4. If proportional symbol, the colour of the symbol selected.

For the choropleth maps, predefined colour schemes are used. The colour schemes are based on ColorBrewer optimal colour schemes from the ColorBrewer website. This limits the user's options for customising the map, but aids in the aesthetics of the map.

The last OGC service used in ThematicWS service is the WMS service standard. This service produces a map styled to the specifications of an SLD style sheet. The parameters in the WMS GetMap request are specified in Table 15 (in Appendix 15).

The interface of the ThematicWS service needs to be designed so that it is easy to use and in line with the other OGC services. The idea is to *extend* the OGC WMSs GetMap request list of parameters. The parameters of the ThematicWS GetThematicMap request are specified in Table 10.

The first 12 parameters are the same as that of the WMS GetMap request to ensure that the user has most of the parameters required to produce a map with the GetMap request. Additional attributes are added at the end of the table (refer to Table 10).

1. ATTRIBUTE

This is a string value corresponding to a specific field name in the attribute table of the dataset. This is a mandatory parameter containing the column name that will be used to classify the data, for example, a census dataset if the user would like to create a map to show the population density of different enumeration areas. In this case, the ATTRIBUTE parameter will be the field population in the dataset.

2. THEMATICTYPE

This is the second mandatory parameter, and can be only one of two values, choropleth or prop\_symbol. If the parameter THEMATICTYPE is equal to choropleth, a choropleth thematic map will be created, and with prop\_symbol, a proportional symbol map is produced.

3. SYMBOL

If the THEMATICTYPE is equal to prop\_symbol, this parameter becomes mandatory. The SYMBOL parameter needs to be set to a well-known symbol type. This is the symbol that will be used to produce the proportional symbol map.

4. SYMBOLCOLOR

If the THEMATICTYPE is equal to prop\_symbol, this parameter becomes mandatory. The SYMBOLCOLOR parameter set the colour of the symbol that was specified in the SYMBOL parameter. The value of this parameter shall be a hex colour code.

5. CLASSED

CLASSED is a mandatory Boolean value, which is either true for classed maps or false for unclassed thematic maps.

6. CLASSIFICATION

If the CLASSED is equal to TRUE this parameter becomes mandatory. In this parameter, the classification method needs to be specified and one of the following options is available: equal\_interval, quantile or mean\_std\_deviation.

7. NROFCLASSES

If the CLASSED is equal to TRUE, this parameter becomes mandatory. This parameter will contain a single integer specifying the number of classes the data should be divided into when a classed map is created.

**Table 10. The parameters of a GetThematicMap request**

Request parameter	Mandatory/optional	Description
VERSION=1.3.0	M	Request version.
REQUEST=GetThematicMap	M	Request name.
LAYERS=layer_list	M	Comma-separated list of one or more map layers.
STYLES=style_list	M	Comma-separated list of one rendering style per requested layer.
CRS=namespace:identifier	M	Coordinate reference system.
BBOX=minx,miny,maxx,maxy	M	Bounding box corners (lower left, upper right) in CRS units.
WIDTH=output_width	M	Width in pixels of map picture.
HEIGHT=output_height	M	Height in pixels of map picture.
FORMAT=output_format	M	Output format of map.
TRANSPARENT=TRUE FALSE	O	Background transparency of map (default=FALSE).
BGCOLOR=color_value	O	Hexadecimal red-green-blue colour value for the background color (default=0xFFFFFF).
EXCEPTIONS=exception_format	O	The format in which exceptions are to be reported by the WMS (default=XML).
ATTRIBUTE	M	A string name of the field.
THEMATICTYPE	M	choropleth/prop_symbol is the available options
SYMBOL (Mandatory if THEMATICTYPE is prop_symbol)	O	Must be a well-known symbol type
SYMBOLCOLOR (Mandatory if THEMATICTYPE is prop_symbol)	O	Must be a hexadecimal color code
CLASSED=TRUE FALSE	M	Should the values divided into classes (default=TRUE)
CLASSIFICATION= method_name (Mandatory if CLASSED is TRUE)	O	Classification method to be used (specify one): equal_interval, quantile or mean_std_deviation
NROFCLASSES (Mandatory if CLASSED is true)	O	The number of classes the data should be divided into.

## 7.6 Implementation of ThematicWS service

Previously in this chapter, the two methods in which the prototype ThematicWS can be implemented were described, a custom orchestration engine and the use of a workflow modeller. This section provides an overview of how the prototypes were implemented using these methods, and the additional services.

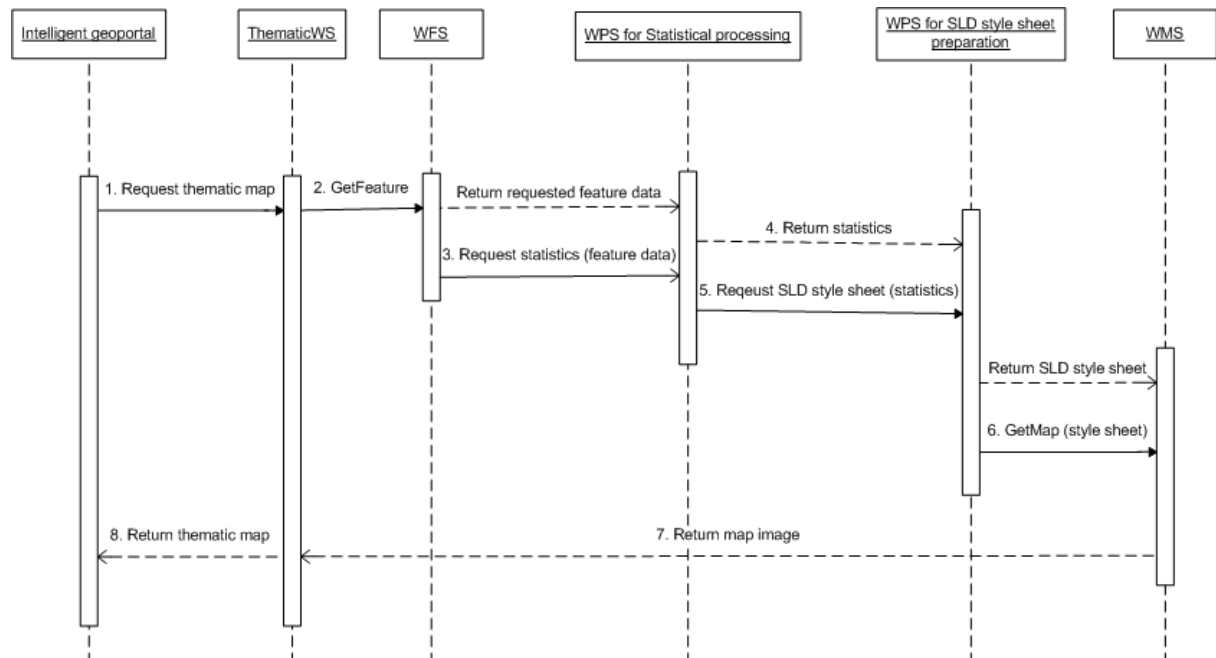
### 7.6.1. Workflow modeller

The prototype was developed using the workflow modeller of the ZOO Project and by wrapping the custom services into two new ZOO WPS services. The ZOO Project was selected from the results obtained in Chapter 8. In this Chapter, the orchestration capabilities of the 52° North and ZOO Project frameworks were evaluated. The results showed that both frameworks have the capability to produce thematic maps using the frameworks workflow modeller.

The ZOO Project was selected in that it allows the user to call external web services, thus allowing the user to call a WMS service, whereas the 52° North framework is very limited in that only WFS and WPS services can be called. Thus, if a WMS service needs to be called within the 52° North workflow modeller, the call would need to be wrapped within a WPS service. The implementation of ThematicWS service with the ZOO Projects' workflow modeller was done with the JavaScript Mozilla SpiderMonkey server, making it easy to call the four services.

The individual WPS services, the statistical processing service and SLD generation service, were both implemented in Java. The reason for this is that 52° North and GeoServer both use Java, and the code could easily be ported to the different applications.

Figure 54 shows the orchestration of the web services to implement the ThematicWS, but during the implementation of ThematicWS with the workflow modellers some difficulties were experienced and the sequence was altered. The sequence diagram in Figure 59 depicts the interaction between the web services for the workflow modeller implementation.



**Figure 59. Sequence diagram illustrating the orchestration of web services in ThematicWS with a workflow modeller**

### **7.6.2. Orchestration engine**

The second prototype was implemented using a custom orchestration engine which is very simplistic in that it is only a service that calls the different services in the correct order and provides them with the required parameters. This differs from the workflow modeller in that all the parameters need to be sent through the chain. This is required, since the workflow modeller is not able to store the parameters. The service ThematicWS was implemented in Java with only one function, GetThematicMap. These functions are explained in Appendix E.

The orchestration engine was implemented using NetBeans integrated development environment (IDE) 7.0.1, the built-in GlassFish server 3.1 and Java Enterprise Edition (EE) 6.

### **7.6.3. Statistical processing and SLD style sheet preparation service**

These services were initially implemented in NetBeans integrated development environment (IDE) 7.0.1, the built-in GlassFish server 3.1 and Java Enterprise Edition (EE) 6, thereafter the services were ported to WPS services ZOO project and the 52° North framework (refer to Chapter 8 for more information).

The statistical processing service performs the standardisation and classification of the data set that is necessary for the production of a thematic map. The statistical processing service consists of 10 functions. An overview of these functions is provided in Appendix D.1.

The SLD style sheet preparation service has two functions, of which one is private (the assignment of colours) and thus cannot be called externally. The public function, getSLD, produces the custom SLD style sheet according to the classes determined by the statistical processing service which is then used by a WMS service to produce the thematic map. Refer to Appendix D.2 for an overview of the functions.

## Chapter 8 Results of an evaluation of the orchestration capabilities of ZOO project and the 52° North framework for an intelligent geoportal

*This section was presented as a peer-reviewed oral presentation 22<sup>nd</sup> International Society for Photogrammetry and Remote Sensing (ISPRS) congress 25 August to 1 September 2012 in Melbourne, Australia, as a paper by Rautenbach, V., Coetzee, S., Strzelecki, M. and Iwaniak, A. under the same title.*

### 8.1. Introduction

A spatial data infrastructure (SDI) is the basis for spatial *data* discovery, evaluation and application for users and providers within all levels of government, the commercial sector, the non-profit sector, academia and by citizens in general (GSDI 2004). The entry point to an SDI is called a geoportal, which typically provides access to spatial data and associated web services in an SDI, facilitating the discovery, display, editing and analysis of data.

A spatial information infrastructure (SII) should provide access to information, i.e. data that has been processed, organized and presented so as to be useful. The geoportal of an SII requires intelligence to orchestrate (automatically coordinate) web services that prepare, discover and present information, instead of data, to the user. The SII's geoportal is referred to as an intelligent geoportal, a geoportal that provides complex functionality through user interface for a user in a specific application domain (Iwaniak et al., 2011).

Standards by the Open Geospatial Consortium (OGC) and the International Organisation for Standardisation's (ISO) technical committee ISO/TC 211, *Geographic information/Geomatics*, facilitate interoperability in a geoportal. The most widely used web service standards in geoportals are the Web Map Service (WMS) and the Web Feature Service (WFS). The Styled Layer Descriptor (SLD) is a visualisation standard from the OGC for specifying user defined styles and the manner in which the WMS shall render the map (OGC, 2007a). The WMS and WFS services, together with SLD, enable the display of spatial data in a geoportal (OGC 2006a, OGC 2010b). Spatial processing can be added to the geoportal with Web Processing Service (WPS) implementations (OGC 2007b). The WPS provides a standardised interface for executing processing services via the Internet.

Interoperability of web services, whether in an SDI or an SII, requires the development and use of uniform standards. Multiple OGC web services are required to prepare and present information in an intelligent geoportal of an SII. Web service orchestration makes it possible for web services to collaborate in a predefined pattern or to be automatically arranged based on local decisions about their interactions with one another at the message and/or execution level (Sun et al., 2010; Suazo &



Aguirre, 2005). Orchestration uses a single process to control and execute the underlying processes in the particular order in which they were specified and assigns the necessary input parameters for the WPS processes. The orchestration process can be altered depending on the output of an individual process. The change is made using the semantic descriptions of the individual processes or services available to the orchestration process.

Cartography methods can be used to create and prepare information from raw data to presentation in the SII's intelligent geoportal. An example of the representation of information is a thematic map, which is used to represent spatial or physical phenomena and the statistical information associated with the particular geographic area (Slocum et al., 2009). Thematic maps have become increasingly popular in the last few years and interest has been shown, especially in the use of web services to produce thematic maps. This can be attributed to the increased use of web services in general, but also in SDIs specifically (Maguire & Longley, 2005).

At the FOSS4G 2011 conference, Garnett and Fenoy presented on a WPS shootout, for which they tested five WPS frameworks: 52° North, constellation, deegree, GeoServer, PyWPS and ZOO project. The shootout evaluated conformance of these frameworks to the OGC WPS 1.0.0 standard and the interoperability of the respective WPS frameworks. All the evaluated frameworks, except the constellation project, passed the tests set out in the shootout. The frameworks we evaluate in this study were included in the shootout and thus are compliant with the OGC standard and provide interoperability. In another study, Lopez-Pellicer et al. (2011) conducted a survey of the availability of WPS services, which showed that out of the 9,329 OGC services discovered on the web, only 58 were WPS services. The survey confirms the OGC implementation statistics (OGC, 2011), which state that the WMS specification is the most implemented of the OGC web services with 46.1% of services discovered. Lack of documentation, profiles and few WSDL descriptions available are some of the limitations that were identified. which create semantic and technical barriers for orchestration of web services.

Business process execution language (BPEL) is an XML-based workflow language that composes services by choreographing service interactions (Weerawarana et al., 2005). BPEL provides control logic operations, event handling, extensibility, and web service policies. Stollberg and Zipf (2007) investigated the efficiency of BPEL and WPS for the orchestration of OGC web services. They found that OGC web services do not support the Simple Object Access Protocol (SOAP) and could therefore not be used in BPEL orchestration engines. Another problem is that orchestration engines could not transfer raw binary data served by the WMS GetMap, for example, thus BPEL could not be used for web service orchestration. A WPS service was successfully used to orchestrate web services in these experiments. OGC has since included SOAP support for their web service standards, eliminating one of the limitations. As a solution to the transfer of raw data, Fleuren and Muller (2008) proposed using a PostGIS database to store intermediate data for orchestration with an ActiveBPEL engine.

In 2010, Foerster et al. stated that composition of web services is still difficult due to interoperability problems. Experiments were done using the 52° North framework to successfully perform map

generalization and transformation. They suggested Content Transformation Services to address the interoperability problems. Workflow modellers emerged to solve the above-mentioned shortcomings of the WPS services. The modellers have since become part of the WPS frameworks that we evaluated.

In this paper we present the results of an evaluation of two frameworks for the orchestration of web services, namely the 52° North geoprocessing framework and the ZOO project. Our evaluation is aimed at comparing the capabilities of each framework for producing thematic maps in an intelligent geoportal. The frameworks were selected for their prominence at conferences and the number of papers written about them. Both frameworks are open source implementations of the OGC WPS standard developed by research companies in Germany and Japan respectively, and have grown to an international user base. The remainder of the paper is structured as follows: first, we provide a brief overview of the WPS standard and present the two orchestration frameworks; next, we describe the evaluation methodology and results; finally, we discuss the results and their implications for SDIs, SIIs, web standards and thematic maps over the Web.

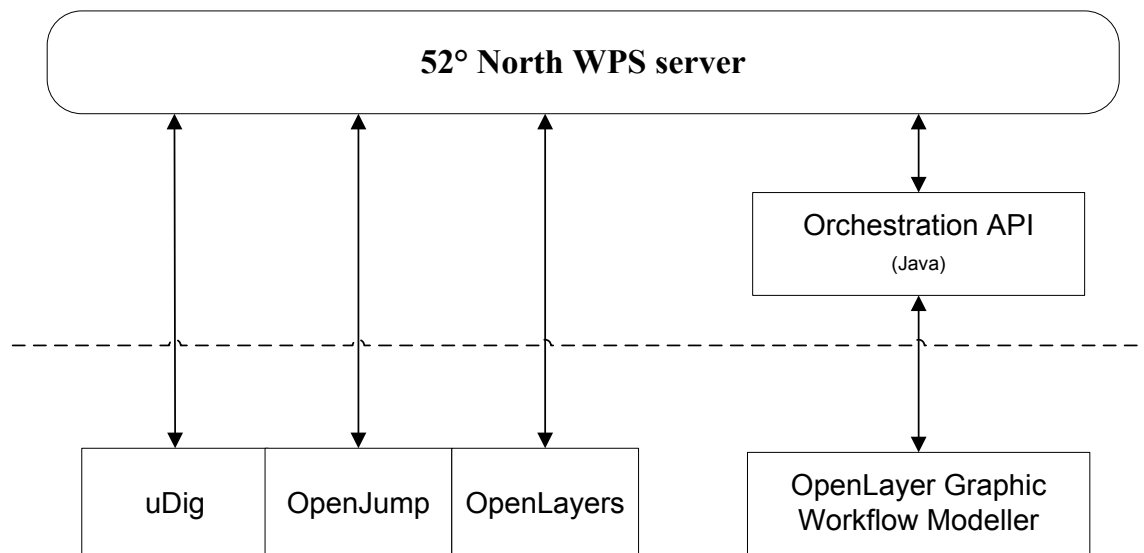
## 8.2. Overview of the WPS standard

The web processing service specification was first released in 2005 as a discussion paper (Foerster & Stoter, 2009). Since then, WPS has become an OGC web service standard with version 1.0.0 currently available (OGC, 2007b). The purpose of the WPS standard is to describe a service that provides spatial data processing functionality which can be executed in a web environment. WPS provides a standardised interface that facilitates the publishing of spatial processes on the web. The OGC WPS standard (2007b) describes a process as any algorithm, calculation or model that operates on spatially referenced data. The standard also describes publishing as the making available of machine-readable binding information and human-readable metadata for service discovery and use. WPS can thus standardize the interface for executing any spatial processes or calculations and provide the process as a web service that can be accessed via the Internet (Fenoy et al., 2010).

At the end of 2011 there were 21 WPS implementations registered with OGC, but no OGC compliant implementations of the WPS standard are available presently (OGC 2011). A survey done in 2011 on the availability of WPS standards showed that most of the WPS implementation was implemented by universities, private companies and government agencies (Lopez-Pellicer et al., 2011). 84% of WPS servers discovered were implemented in Europe, with universities as the implementer. A workflow modeller is in some cases implemented together with the WPS implementation. According to Fenoy et al., some of the most popular open source WPS implementations are 52° North, ZOO project, deegree, WPSint and PyWPS (2010).

### 8.3. Overview of WPS frameworks

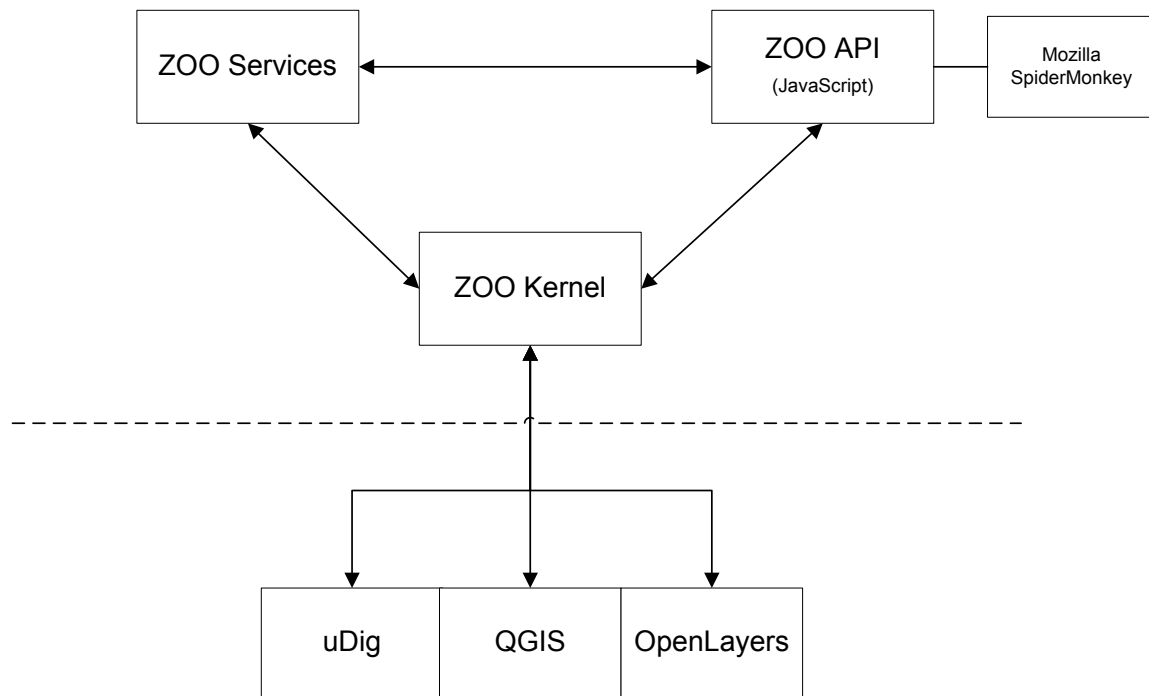
52° North framework, founded in 2004, is an open source framework that enables the deployment of spatial processes on the web in a standardised way (<http://www.52north.org/wps>). 52° North framework is a Java based pluggable framework that offers spatial processes via the OGC WPS interface (Baranski, 2008). WPS processes can be created using standard Java libraries and other 3rd party libraries. The 52° North framework has a web administration tool that can be used to upload WPS processes, connect to a repository of WPS processes, and to adjust server settings.



**Figure 60.** Depicts the architecture of the 52° North framework

52° North framework provides a WPS orchestration API and a graphical modelling tool for geoprocessing workflows based on the orchestration API (refer to Figure 60). The modelling tool, known as the WPS Workflow Modeller, allows visual modelling of geoprocessing workflows in an Internet browser environment (<http://52north.org/communities/geoprocessing/workflows/index.html>). The workflow modeller allows the programmatic chaining of WPS services and complex data inputs from, for example, other OGC services such as WFS.

The ZOO project, founded in 2008, is an open source project that provides a developer friendly WPS framework for the development of WPS services (<http://www.zoo-project.org/>). The ZOO project consists of three main components, namely (refer to Figure 61): the ZOO kernel, ZOO services and the ZOO API. The ZOO kernel is a server-side C kernel, which forms the core of the ZOO project and allows the creation, management and chaining of WPS 1.0.0 compliant web services. The kernel supports a wide variety of programming languages: C, Python, Java, JavaScript, Fortran and PHP, and can easily be connected to external spatial libraries, scientific models, cartographic engines and spatial databases. The large variety of languages is convenient for developers, allowing the re-use of existing code to create new web services with the ZOO project.



**Figure 61. Depicts the architecture of the ZOO framework, adapted from ZOO Project**

The ZOO service provider consists of the Service Shared Object (SSO) and a metadata ZOO configuration file (zcfg) per provided service (Fenoy et al., 2010). This facilitates the GetCapabilities and DescribeProcess request specified in the WPS 1.0.0 standard specification. The kernel parses the zcfg file using Flex and Bison.

The ZOO API is a server-side JavaScript library designed to simplify the WPS process creation and chaining (Fenoy et al., 2010). The API provides ready-to-use JavaScript functions for handling WPS requests via HTTP. Orchestration of WPS services is possible with the API through the use of a specific method. The method also provides the ability to add a level of control and logic to the WPS chaining. The proj4js library is incorporated into the API, allowing server-side re-projection of spatial data.

The frameworks are platform independent, but it can be deduced from the available installers, ease of installation on the different frameworks, and the operating system used in the documentation that the ZOO framework is designed originally for Linux and the 52° North framework for Windows.

## 8.4. Evaluation and results

For our evaluation of the suitability of the 52° North and ZOO Project frameworks for thematic maps, we included the following:

1. Documentation and support available.
2. Installation of the framework.
3. Support for protocols, such as, RESTful, SOAP and HTTP GET/POST.

4. Process management capabilities, e.g. whether user interference is possible once the orchestrated process has been started.
5. Workflow construction capabilities.
6. Support for asynchronous processing.
7. Capacity to provide and use semantic descriptions.
8. Data transport mechanism, e.g. whether the data is included in the request or sent pointed to by a URL.
9. The flexibility of the parameters, e.g. whether are parameters required and whether additional parameters can be added.
10. Ease of extending the framework, e.g. the ability to add additional functionalities required.
11. Ease of integration with other GIS applications.
12. Support for the orchestration of external OGC services, e.g. calling a GeoServer WFS to obtain attribute data.

For our evaluation of the frameworks, we orchestrated OGC standard web services and components to produce thematic maps. The orchestration process is based on experiments we are currently busy with. Our current findings show that on both frameworks it is possible to orchestrate a thematic web service from standard components, such as WMS, WFS and WPS, that make use of other standards, such as SLD and Common Query Language (CQL). Some customised programming is required, such as the classification of the dataset and generation of SLD. These processes need to be wrapped into WPS services that make the orchestration of standard web service components possible. The evaluation was performed on a computer running Mac OS X Lion 10.7.2 with Xcode 4.2. The ZOO project version 1.2, 52° North WPS 2.0 RC7 and 52° North workflow modeller version 1.0.0 released December 2011 was used for this evaluation. Table 11 shows the results of our evaluation. A discussion of these results follows in Section 8.5.

**Table 11. Comparison of 52° North and ZOO Project framework according to the criteria specified**

Criteria	52° North WPS framework	ZOO Project
Documentation and support available	Documents and tutorials are available online; however, they are outdated and refer to old versions of the software. An online forum and mailing list are available and a post to forum was answered quickly and very proficiently. Documentation mainly covers Windows OS.	Documents and tutorials are available online in the following formats: HTML, epub and pdf. Documentation is up-to-date and detailed. The available tutorial is aimed at OSGeo live disk (Linux). An online forum and mailing list are available and a post to forum was answered quickly and very proficiently.
Installation of the framework	An installer is available for Windows, with an embedded version of Tomcat. For Linux and Mac a WAR file must be deployed in a Tomcat server container.	The installation of ZOO is not simple. An installer is available for Mac, but still required additional libraries. In our experience it was a daunting process to obtain the library dependencies and correct versions in

		order to compile the framework.
Support for protocols	Framework support for SOAP and HTTP GET/POST. Exposes the service as a WSDL document.	Framework support for HTTP GET/POST. SOAP support is currently not implemented in the framework (except for WPS requests and responses that can be packed into a SOAP:Envelope).
Process management capabilities	No mechanism for process intervention is currently implemented, but can easily be supported through the use of signals provided by the inter process communication (IPC), such as PAUSE and QUIT. Status reporting is supported through shared memory messaging.	No mechanism for process intervention is currently implemented, but can easily be supported through the use of signals provided by the inter process communication (IPC), such as PAUSE and QUIT. Status reporting is supported through shared memory messaging.
Workflow construction capabilities	A graphic user interface (GUI) is available for implementing the orchestration API, but offers no control logic operations, e.g. if and while.	The ZOO API is a JavaScript API that runs server-side. It provides control logic operators and the functionality for web service chaining.
Support for asynchronous processing	Synchronous and asynchronous invocation.	Synchronous and asynchronous invocation.
Capacity to provide and use semantic descriptions	The user can create a process description file, containing information, such as title and abstract, can create an optional process description with data inputs and process outputs. The response of the DescribeProcess request is constructed from the information in the description file.	Each service must have a zcfg configuration file, which contains some metadata: title, abstract, process version, service type and information on the input and output variables, to name a few. The response of the DescribeProcess request is constructed from the information in the configuration file.
Data transport mechanism	A literal can be sent via the URL as a value or pointed to using an URL within the request URL.	mimeType (multipurpose internet mail extension) are used, especially the JSON type for complex data.
The flexibility of the parameters	Parameters can be set to be mandatory or optional, and multiple occurrences of the same variable are possible.	Parameters can be set to be mandatory or optional, and multiple occurrences are possible.
Ease of extending the framework	The source code of the entire project is available in a revision control system. Additional tutorials are available on how to extend the framework, making this easy for a novice to understand.	The source code of the entire project is available in a revision control system. There is no tutorial on how to extend the framework.
Ease of integration with other GIS applications	Plugins are available for uDig, OpenJUMP and OpenLayers to provide them with the capability of calling a 52° North WPS process. Extensions for WPS4R (an R backend), GRASS, SEXTANTE and ArcGIS connections are	Plugins are available for uDig, QGIS and OpenLayers to provide them with the capability of calling a ZOO WPS process. Extensions for GDAL/OGR, GRASS and MapServer connections are available, allowing the implementation of a new process

	available, allowing the implementation of a new process using their functionalities.	using their functionalities.
Support for the orchestration of external OGC services	External WMS, WFS and WPS services can be added to the map in the workflow GUI. However, the WFS and WPS can only be used as input to a WPS and the WMS only provides base maps and additional layers to the output map.	External OGC web services can be called in the ZOO API (JavaScript) with GET and POST requests.

## 8.5. Discussion of results

Our goal was to evaluate the 52° North and ZOO frameworks capabilities for the orchestration of web services to produce thematic maps. The 52° North framework is an easy-to-use WPS framework with a graphic modelling tool, outstanding for beginners. The documentation available is excellent for novices, but more expert documentation is required. The ZOO framework provides support for multiple programming languages, which allows the re-use of existing code to create new web services with the ZOO project making it convenient for developers. The documentation is extensive and describes most aspects of the framework in elaborate detail.

The difference in ease of installation between the two frameworks was clear. This could be due to the fact that 52° North is a more mature project which was started four years prior to the ZOO project. Documentation for both frameworks is available; however, the 52° North documentation is outdated and refers to old versions of the software and library dependencies.

Both frameworks currently support HTTP GET/POST and the SOAP protocol. 52° North exposes its WPS services as WSDL document. The ZOO project does not support WSDL for the GetCapabilities request. However, this is presently an open ticket classified as a major defect in the Trac project management system used by ZOO.

The 52° North framework provides a GUI workflow modeller, which is easy to use and requires no programming experience. The workflow modeller allows the user to create an orchestrated service from the available web services. WPS services from the framework, as well as external WPS and WFS services, can be called, e.g. GeoServer services, and orchestrated in the workflow modeller. The graphic modeller (<http://giv-wps.uni-muenster.de:8080/geoserver/ows>) has an easy to interpret GUI with minimal options making it user-friendly. Literal data nodes can be created and linked to the specific web service. The modeller provides the choice to execute a single WPS service or service chain. The orchestration API written in Java is also available for users, but this requires some programming expertise. The ZOO project has no GUI modeller, but a server-side JavaScript API that runs on SpiderMonkey, called ZOO API, for web service orchestration. The ZOO API gives the power of JavaScript to the user allowing them to call any web service via HTTP and use control logic operations. The JavaScript ZOO API provides more functionality to the user than the 52° North workflow modeller, but requires a certain level of JavaScript skill.

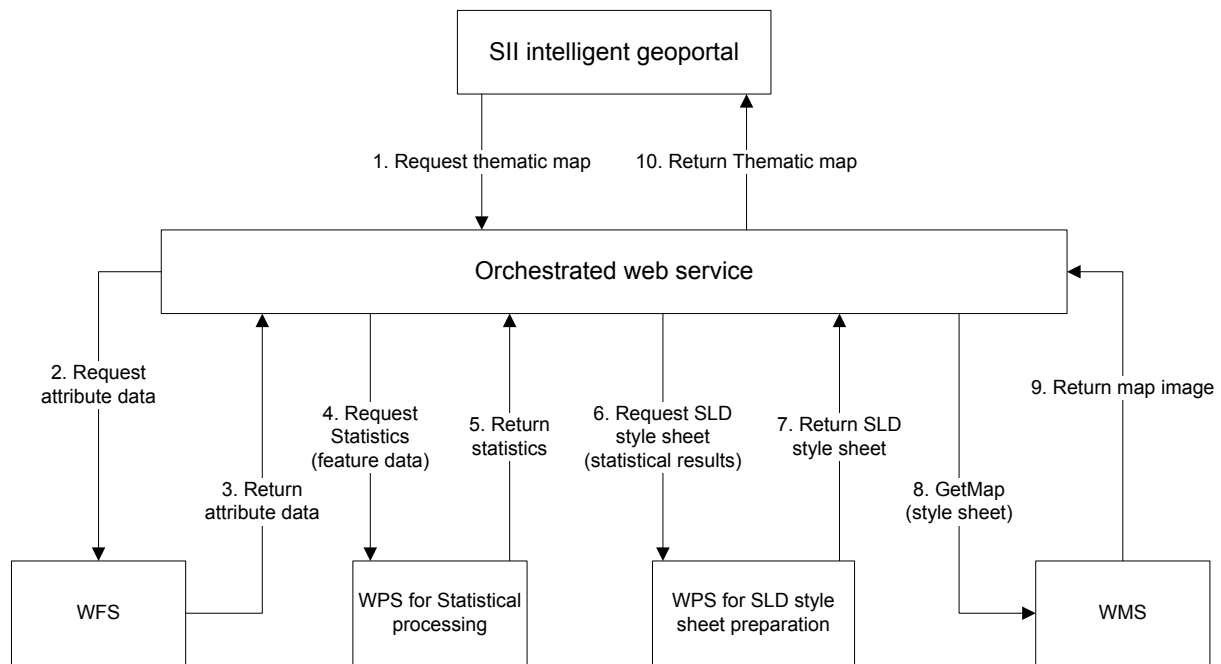
The ease of extendibility of the framework is an important consideration for the developers. This allows them to customize the framework to suite their requirements. Both frameworks are open source, and thus the source code can be examined and change by the user in accordance with the licence. The evaluated frameworks both have well-structured source code that can be obtained from a revision control system. The community forums provide support for users and developers.

At present the 52° North orchestration API and ZOO API do not comply with any standard such as BPEL. The 52° North framework developers are busy developing a BPEL workflow modeller to rectify this limitation. The ZOO API uses the well-known JavaScript and thus provides additional functionalities for calling the ZOO WPS services. There are mentions of creating a graphic workflow modeller similar to YAWL (<http://www.yawlfoundation.org/>), which is a business process modeller with native data handling for XML, XQuery and XPath. A current limitation to automatic orchestration of web services in an intelligent geoportal of the SII is the lack of semantic information available. Both frameworks are excellent for orchestration of web services in a predefined pattern; nevertheless limitations occur when decisions have to be made in run-time according to the output of a previous process. At present, the frameworks provide basic semantic information: title, abstract, list of input and output literals. This information is used to construct the response of the GetCapabilities request. Additional semantic information for the automatic orchestration of the available web services is required so that the next process can be selected based on the semantic information and ontologies, providing semantic interoperability. Semantical interoperability is used to ensure that the contents of the data and the service are correctly understood when the components are connected (Yue et al., 2007).

Figure 62 depicts the orchestration process of web services to produce thematic maps. The orchestration with 52° North could not perform the last step of calling a WMS from GeoServer WMS that produce the resulting thematic map. The 52° North development team are planning to extend the functionalities of the workflow modeller to allow the orchestrated WMS. The rest of the orchestration process in our experiment could be performed with the current functionalities of the 52° North WPS framework. With the ZOO framework, the entire orchestration process to produce thematic maps could be performed. The ZOO API, which relies on JavaScript, provides developers with the ability to call external OGC web services and to use control flow logic operators.

Results of the evaluation show that both frameworks have potential to facilitate orchestration in an intelligent geoportal, but that some functionality is still lacking. The orchestration of web services allows the development of more advanced web services for the preparation, discovery and presentation of information. The results of our evaluation of these frameworks, both with their respective strengths and weaknesses, can guide developers to choose the framework best suitable for their specific needs and requirements.





**Figure 62. A sequence diagram depicting the orchestration of web services to create thematic maps**

## 8.6. Conclusion

In this paper we presented our evaluation of the orchestration capabilities of the 52° North and ZOO project framework for producing thematic maps. In our experiments we showed that there are limitations and that they differ for each framework. A limitation both frameworks have is the lack of semantic information needed for automatic orchestration.

The functionality of the framework is very important, but even if these functionalities are excellent, it is important to have up-to-date documentation and an easy to install framework. A cumbersome installation creates a barrier, resulting in only extremely dedicated and technically proficient developers using the framework. This barrier also prevents the use of the framework by a larger community. This concern was reinforced by the results of the Lopez-Pellicer et al. (2011) study, which identified that the top implementers of WPS are Universities. From these results it can be deduced that WPS implementations are still an academic exercise and that the service is not being used by the industry.

In further work we aim to evaluate the performance of the different frameworks. We plan to compare their performance with other workflow software available.

The development teams of the evaluated frameworks are addressing some of these limitations that are identified in the paper and we are excited to see the future of these frameworks. We hope our research will contribute to the use and improvement of workflow modeller for the orchestration of web services.

## Chapter 9 Discussion of results

The results obtained from the experiments and implementation of ThematicWS are discussed in this chapter. The goal of this project was to investigate how standard OGC web services can be orchestrated in order to produce thematic maps. In Chapter 1 (refer to Section 1.3) the objectives of the project are defined and in this section the results obtained in regards to each objective is explained.

A literature review was performed to investigate existing theories and related work in the fields of thematic cartography, OGC standards and web services, spatial data infrastructure, spatial information infrastructure, geoportals, and other related work. The literature survey was of great importance since the project is cross disciplinary and requires an understanding of both cartography and computer science. The literature survey can be found in Chapter 2 to 5 of this dissertation.

In Chapter 6 (refer to Section 6.3) the model for the process of producing choropleth and proportional symbol maps was defined. The model provides a step-by-step process for producing the thematic maps according to cartographic rules. These steps can now be transformed into algorithm or programming code for the development of a thematic web service.

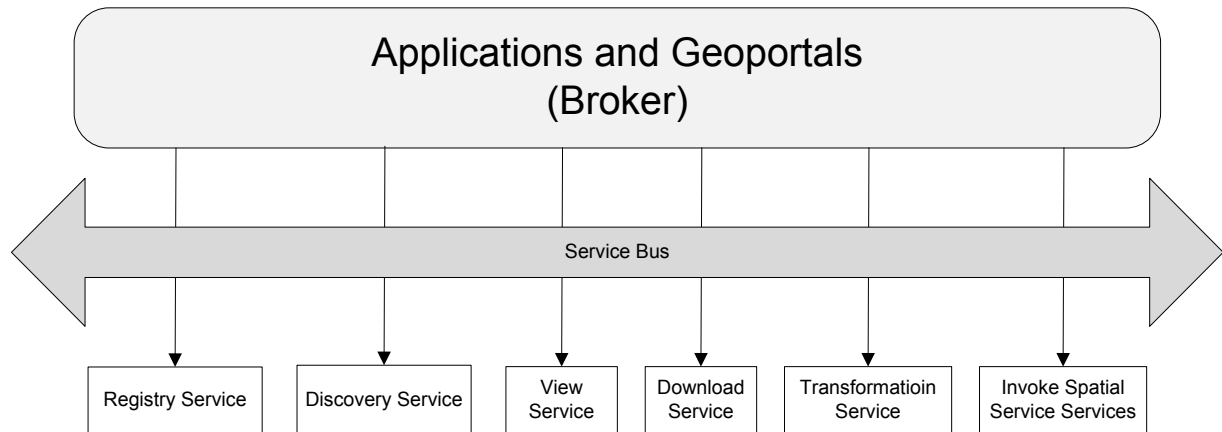
The model was then used to determine the required services for the development of ThematicWS, a WFS service to obtain the attribute data, statistical processing service to perform standardisation and classification, SLD generator service to create the style sheet, and a WMS service to produce the final thematic map. The WFS and WMS services are standard services that have been defined by OGC. GeoServer implementations of these services were used to develop the ThematicWS service. The use of SLD in GeoServer allows the use of the SLD extensions that were implemented in GeoServer (refer to Section 3.5.3). The extensions reduce some redundancy in the SLD style sheet; however, since these extensions are not standard, relying on them will limit interoperability.

There are no standard services available for the statistical processing and SLD generation; however, these custom services can be wrapped in WPS services, which provides a standard interface to the custom functionalities required. When using a WPS service for the custom functionalities, all the benefits of using standard services are gained. The lack of standard services for statistical processing and SLD generation causes difficulties for on-the-fly orchestration, since the interface for custom services can vary (Rautenbach et al., 2012a). However, when the custom services are wrapped in a WPS service, the interface is now standard and known, which makes the orchestration of the WPS service with other standard services easier, and allows for interchangeability and interoperability.

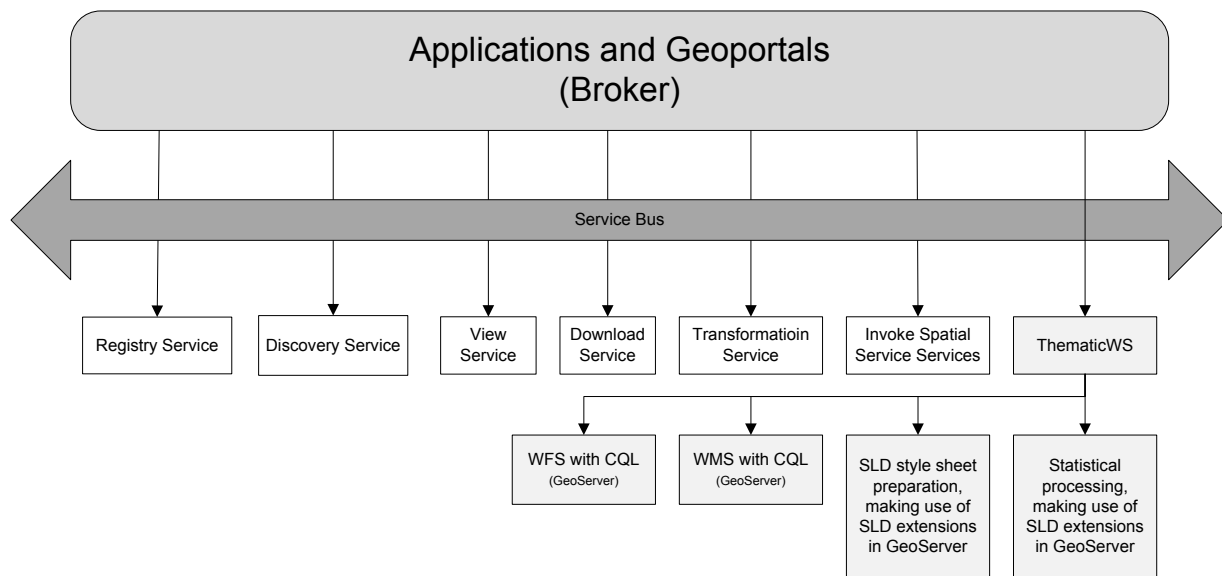
One of the requirements of the ThematicWS is that the service should be easily integrated into a geoportal of an SDI. The functionality in a geoportal is provided by services that allow the user to discover, edit and analyse spatial information. Figure 63 shows the architecture of the conventional SDI and geoportal. It can be seen that the data and services can be accessed through a broker, which

can be a geoportal. The services consumed by geoportals are usually atomic non-complex services, which provide a single function.

The ThematicWS service is intended to function in an intelligent geoportal (introduced in Section 4.4) that makes use of complex orchestrated web service to provide functionalities for the users. Figure 64 shows that the service is composed from existing standard components, WFS, WMS and WPS.



**Figure 63. Current SDI geoportals with available service types, adapted from the INSPIRE Network Service Architecture (2007)**



**Figure 64. Components of the ThematicWS (Rautenbach et al., 2012a)**

A number of WPS implementations are available, such as 52° North, ZOO project, deegree, and GeoServer WPS. From previous publications and conference proceedings it was clear that 52° North and ZOO project's WPS implementations are two of the more popular implementations of the OGC WPS standard. These frameworks also implement a workflow modeller that can be used to

orchestrate services. In Chapter 8 the findings of an evaluation on these frameworks can be found. The results showed that both frameworks are capable of orchestrating the services required for ThematicWS. An additional finding showed that the framework's ease of use might create barriers for novice users and the wide spread development and use of WPS services (Rautenbach et al., 2012b).

The ThematicWS service was implemented using two methods: making use of a workflow script, and workflow modellers. For both methods the custom functionality within the WPS services were developed in Java, since Java is supported by both 52° North and the ZOO project. Both the workflow script and workflow modeller implementation method of the ThematicWS service were found to be successful. The workflow script was also developed using Java and would call and handle the variable as required. As for the workflow modeller, the ZOO project was selected as a result of the evaluation performed in Chapter 8. The ZOO project uses server-side JavaScript to perform workflow modelling, which was found to be very easy to use since JavaScript is a simple well-known scripting language and provides all the required functionality for the orchestration.

The testing interface for both the implementation was a simple HTML page, the parameters could be entered to request a thematic map. Appendix C provides examples of respective request and response of thematic maps produced by the ThematicWS service (the request parameters are explained in Table 10).

## Chapter 10 Conclusion

### 10.1. Introduction

The work in this dissertation investigated whether an orchestrated service can be developed from standard services to produce thematic maps. The final chapter provides both an overview of the main results and information on future work that still needs to be addressed in this field.

### 10.2. Main results from this dissertation

In this dissertation the results of an investigation on how standard OGC web services can be orchestrated to produce thematic maps are shared. The results showed that it is possible to orchestrate OGC standard services to produce choropleth and proportional symbol maps.

A model of the thematic cartographic process for the development of choropleth and proportional symbol maps were developed. The model was developed using a review of theoretical cartography, best practices, and an evaluation of implementations of thematic mapping. A set of sequence steps and flow diagrams are used to depict the model. The model can be transformed into an algorithm that can be used to develop the thematic web service.

The evaluation of the orchestration capabilities of ZOO project and the 52° North framework provides information for both developers of the frameworks and users. For developers the evaluation provides information on possible shortcomings and barriers for novice users that need to be addressed. The results provide a guide to users on the frameworks and aids in selecting an appropriate framework for their requirements.

From the developed model it was determined that the thematic cartographic process for producing choropleth and proportional symbol maps can be divided into four services: WFS for retrieving the attribute data, statistical processing service, SDL generator service, and a WMS service to produce the thematic map. In order to orchestrate only standard service, the custom services need to be wrapped in WPS services to provide a standard interface for the orchestration. The WPS wrapping of custom services proved to be an adequate method when orchestrating services. Using WPS services removes on-the-fly orchestration difficulties, and provides interchangeability and interoperability between the different services in the implementation of orchestrated services, such as ThematicWS.

The ThematicWS service was implemented using two distinct methods: a workflow script and workflow modeller. Both these methods were successful in orchestrating the standard services (WMS, WFS, and WPS) for producing thematic maps.

The functionalities of a geoportal are provided through the use of web service such as the WFS for viewing and editing the attribute data. Previous thematic mapping solutions have relied on extending

SLD for thematic mapping; however, the users were still required to manually develop the SLD style sheet. In this solution the SLD is automatically created by the SLD generator service according to the results of the statistical processing. The four required services are then coordinated and orchestrated by the controller service (ThematicWS), which can be easily called and consumed by the geoportal of an SDI to produce thematic maps according to user defined parameters.

### **10.3. Recommendations for further research**

The WPS framework evaluation (refer to Chapter 8) needs to be performed on more WPS and workflow implementations. The performance of the frameworks as well as the data handling capabilities should also be investigated further.

The semantic metadata required for automated orchestration of web services should be investigated in more detail. It is also necessary to investigate whether any refinements to existing SDI web service discovery mechanisms are required to enable the automated discovery of services. Semantic services and interoperability are briefly discussed in Section 3.4 and 5.5. However, a review of the state-of-the-art is required in semantic interoperability and services.

The current implementation of the ThematicWS service is limited to choropleth and proportional symbol maps. It should be investigated whether this method could be used for the other thematic cartographic map types, and address the issues that may occur. Developing a semantic thematic mapping service that will automatically select and orchestrate the required services for this semantics needs to be employed.

Developing a metadata schema that describes thematic maps, which will provide cartographic information interoperability is needed. Such a schema will enable the overlay of different thematic maps (e.g. using different classification methods and symbolization) showing sea water temperatures in different years.

## References

1. Alameh N (2003). Service chaining of interoperable geographic information web services. *Internet Computing*, 7(1), pp. 22-29.
2. Baranski B (2008). Grid Computing Enabled Web Processing Service, *GI Days 2008*, Muenster, Germany, 16-17 June 2008.
3. Bernard L, Kanellopoulos I, Annoni A and Smits P (2005). The European geoportal—one step towards the establishment of a European Spatial Data Infrastructure, *Computers, Environment and Urban Systems*, Volume 29, pp. 15–31.
4. Cardoso J (2006) Approaches to Developing Semantic Web Services, *International Journal of Computer Science*, Volume 1(1), pp. 8-21.
5. Cardoso J and Sheth A (2005) Introduction to Semantic Web Services and Web Process Composition. In *Semantic Web Process: powering next generation of processes with Semantics and Web Services*, Lecture Notes in Computer Science, Springer.
6. Centre for Spatially Integrated Social Science (2003). GeoDa User's Guide, available online at <http://geodacenter.org/downloads/pdfs/geoda093.pdf>, accessed 3 July 2011.
7. Čerba O (2010). *XSLT Templates for Thematic Maps*, In Lecture Notes in Cartography in Central and Eastern Europe, by Gartner G and Ortog F, pp. 181-190, Springer, New York, 2010.
8. Clarke D (2011). Vision for a spatial data infrastructure in South Africa. *Spatial data infrastructure workshop*. Cape Town, South Africa. May 2011, available online at [http://web.up.ac.za/sitefiles/file/48/16053/Clarke\\_2011\\_SDIWorkshop\\_VisionForSASDI.pdf](http://web.up.ac.za/sitefiles/file/48/16053/Clarke_2011_SDIWorkshop_VisionForSASDI.pdf) accessed 24 October 2011.
9. Coetzee S (2008). *An analysis of a data grid approach from spatial data infrastructures*, PhD dissertation, University of Pretoria, South Africa.
10. Coetzee S (2011). Reference model for a data grid approach to address data in a dynamic SDI, *GeoInformatica*, Springer, May 2011, available online at <http://www.springerlink.com/content/px350m5045084814/>, accessed 25 June 2011.
11. Councill WT and Heineman GT (2001). *Definition of a Software Component and its Elements*. In *Component-based Software Engineering*, by GT Heineman and WT Councill, pp. 5–20, Boston: Addison-Wesley.

12. Craglia M (2010). *Building INSPIRE: The Spatial Data Infrastructure for Europe*, Joint Research Centre of the European Commission, available online at <http://www.esri.com/news/arcnews/spring10articles/building-inspire.html>, accessed 25 June 2011.
13. Crampton JW (2009). *Rethinking maps and identity*, In *Rethinking Maps*, by M Dodge and R Kitchin, C Perkins, pp.26-49, Abingdon: Routledge, 2009.
14. Cuff D and Mattson M (1982). *Thematic Maps: Their Design and Production*, New York: Methuen & Co.
15. Danapaquame N and Bhavani P (2010). A Novel Approach for Parallel Web Service Composition, *International Conference on Communication and Computational Intelligence*, 27 – 29 December 2010, pp. 613-616, India: Kongu Engineering College.
16. Dent B (1996) *Cartography: Thematic Map Design*, 4<sup>th</sup> edition, William C Brown Publishers.
17. De Longueville B (2010). Community-based geoportals: The next generation? Concepts and methods for the geospatial Web 2.0, *Computers, Environment and Urban Systems*, Volume 34, pp. 299–308.
18. Dietze L and Zipf A (2007). Extending OGC Styled Layer Descriptor (SLD) for Thematic Cartography, *4th International Symposium on LBS and Telecartography*, 8-10 November 2007, Hong Kong, China, available online at <http://www2.geoinform.fh-mainz.de/~zipf/Thematic-SLD.LBS-Telecarto2007.pdf>, accessed on 15 March 2011.
19. Donker FW (2009). *Public Sector Geo Web Services: Which Business Model Will Pay for a Free Lunch?*, In *SDI Convergence: Research, Emerging trends, and Critical Assessment* by B. van Loenen, JWJ Besemer and JA Zevenbergen, pp. 35-51, Optima Grafische Communicatie: Delft, The Netherland, 2009.
20. ESRI (2010). *ArcGIS: A Complete Integrated System*, available online at <http://www.esri.com/software/arcgis/index.html>, accessed 3 July 2011.
21. Fenoy G, Bozon and Raghavan V (2010). ZOO Project: The open WPS platform. WebMGS 2010, *1st International workshop on pervasive web mapping, geoprocessing and services*, 26-27 August 2010, Politecnico di Milano, Como, Italy.
22. Fleuren T, Götze J, Droanca V and Müller P (2010). Improving Data Management of Orchestrated Web Services, *Proceedings of the Work in Progress Session of Software Engineering and Advanced Applications*, available online at <http://dSPACE.icsy.de:12000/dSPACE/handle/123456789/292>, accessed 16 March 2012.



23. Fleuren T and Muller P (2008). BPEL Workflows Combining Standard OGCWeb Services and Grid-enabled OGC Web Services. *Software engineering and advanced applications, SEAA Conference*, 3-5 September 2008, Parma, Italy.
24. Foerster T, Lehto L, Sarjakoski T, Sarjakoski L and Stoter J (2010). Map generalization and schema transformation of geospatial data combined in a Web Service context. *Computers, Environment and Urban Systems*, Volume 34, pp. 79-88.
25. Foerster T and Stoter J (2009). Establishing an OGC Web Processing Service for generalization processes, *Workshop of the ICA Commission on Map Generalisation and Multiple Representation*, 25 June 2009.
26. Fu P and Sun J (2010). *Web GIS principles and applications*. ESRI Press.
27. Garnett J and Fenoy G (2011). WPS Shootout. *FOSS4G*, 12-16 September 2011, Denver, United States of America.
28. GeoServer (2011). *Styling*. Available online at <http://docs.geoserver.org/stable/en/user/>, accessed 27 June 2011.
29. Gone M and Schade S (2007). Towards Semantic Composition of Geospatial Web Services – Using WSMO in Comparison to BPEL, *GI Days 2007*, Muenster, Germany, 10 – 12 September 2007.
30. GSDI (2004). *The SDI Cookbook*, Edited by Nebert D, 25 January 2004, available online at <http://www.gsd.org/docs2004/Cookbook/cookbookV2.0.pdf>, accessed 20 July 2011.
31. Haklay M (2010). *Interacting with Geospatial Technologies*, John Wiley & Sons: Chichester.
32. He X, Persson H and Östman A (2011). Geoportal usability evaluation, *International Journal of Spatial Data Infrastructures Research*, submitted 31 May 2011, available online at <http://ijmdir.jrc.ec.europa.eu/index.php/ijmdir/article/view/248/297>, accessed 19 February 2012.
33. Hong J and Lin S (2005). Web-based Thematic Map Service in OpenGIS Environment, *The 26th Asian Association on Remote Sensing*, 7-11 November 2005, Hanoi, Vietnam.
34. Huang Z, Li G, and Zeng Y (2007). Building data infrastructure for geo-computation by spatial information grid, *Proceedings of the Sixth International Conference on Grid and Cooperative Computing*, IEEE Computer Society, Washington, DC, available online at <http://portal.acm.org/citation.cfm?id=1303742>, accessed 25 July 2011.
35. INSPIRE (2007). *INSPIRE Network Service Architecture*, available online at [http://inspire.jrc.ec.europa.eu/reports/ImplementingRules/network/D3\\_5\\_INSPIRE\\_NS\\_Architecture\\_v3-0.pdf](http://inspire.jrc.ec.europa.eu/reports/ImplementingRules/network/D3_5_INSPIRE_NS_Architecture_v3-0.pdf), accessed 5 July 2011.

36. INSPIRE (2011). *Spatial Data Infrastructures in Europe: State of play spring 2011*, 6 July 2011, K.U. Leuven publishers, available online at [http://inspire.jrc.ec.europa.eu/reports/stateofplay2011/INSPIRE\\_NSDI\\_SoP\\_-\\_Summary\\_Report\\_2011\\_-\\_v6.2.pdf](http://inspire.jrc.ec.europa.eu/reports/stateofplay2011/INSPIRE_NSDI_SoP_-_Summary_Report_2011_-_v6.2.pdf), accessed 19 February 2012.
37. Iosifescu-Enescu I, Hungentobler M and Hurni L (2007). Cartographic extensions to OGC Web Map Services, *23rd International Cartographic Conference*, 4-10 August 2007, Moscow, Russia.
38. Iosifescu-Enescu I, Hungentobler M and Hurni L (2010). Web cartography with open standards – A solution to cartographic challenges of environmental management, *Environmental Modelling & Software*, Volume 25, pp. 988–999.
39. Iwaniak A, Kaczmarek I, Kubik T, Lukowicz J, Paluszyński W, Kourie D, Cooper A and Coetzee S (2011). An Intelligent Geoportal for Spatial Planning, *25th International Cartographic Conference*, 4-8 July 2011, Paris, France.
40. Iwaniak A and Kubik T (2010). Building and maintaining metadata repositories with the aid of ontology tools and technologies, *GSDI 12*, 19 – 22 October 2010, Singapore, Singapore.
41. Ioup E, Lin B, Sample J and Shaw K (2008). *Geospatial Web Services: Bridging the Gap between OGC and Web Services*, In *Geospatial Services and Applications for the Internet*, by Sample J, Shaw K, Tu S and Abdelguerfi M, pp.73 - 93, New York, Springer, 2008.
42. Janowicz K, Schade S, Bröring A, Keßler C, Maué P and Stasch C (2010) Semantic Enablement for Spatial Data Infrastructures, *Transactions in GIS*, Volume 14, pp. 111-129.
43. Jordan M (2010). *Dynamic Symbolizers*, GeoServer, available online at <http://blog.geoserver.org/2008/12/08/dynamic-symbolizers-part-1/>, accessed 25 June 2011.
44. Katragadda L (2009). Making maps to fight disaster, build economies, *TED Conference*, November 2009, Mysore, India, available online at [http://www.ted.com/talks/lalitesh\\_katragadda\\_making\\_maps\\_to\\_fight\\_disaster\\_build\\_economies.html?awesm=on.ted.com\\_613V](http://www.ted.com/talks/lalitesh_katragadda_making_maps_to_fight_disaster_build_economies.html?awesm=on.ted.com_613V), accessed on 25 July 2011.
45. Koblin A (2011). Artfully visualizing our humanity, *TED Conference*, February 2011, Long Beach, California, available online at [http://www.ted.com/talks/lang/eng/aaron\\_koblin.html](http://www.ted.com/talks/lang/eng/aaron_koblin.html), accessed on 25 July 2011.
46. Lopez-Pellicer FJ, Rentería-Agualimpia W, Béjar R, Muro-Medrano PR and Zarazaga-Soria FJ (2011). Availability of the OGC geoprocessing standard: March 2011 reality check. *Computers & Geosciences*, available online at <http://www.sciencedirect.com/science/article/pii/S009830041100358X>, accessed 17 February 2012.

47. Ludwig B and Coetzee S (2010). A comparison of platform as a service (PaaS) clouds with a detailed reference to security and geoprocessing services, WebMGS 2010, *1st International workshop on Pervasive Mapping, Geoprocessing and Services*, 26-27 August 2010, Lake Como, Italy.
48. Maguire DJ and Longley PA (2005). The emergence of geoportals and their role in SDI, *Computers, Environment and Urban Systems*, Volume 29, pp. 3–14.
49. Masser I (2005). *GIS worlds: Creating spatial data infrastructures*, ESRI Press: Redlands, CA, 2005.
50. Nedović-Budić Z, Cromptvoets J and Georgiadou Y (2011). *Spatial data infrastructures in context*, CRC Press, Taylor & Francis, Boca Raton, 2011.
51. Olivier MS (2004). *Information Technology Research. A Practical Guide*, Pretoria, Van Schaik Publishers, 2004.
52. Open Geospatial Consortium (2007). *Discussions, findings, and use of WPS in OWS-4*, Edited S Keens, 6 June 2007, available online at [http://portal.opengeospatial.org/files/?artifact\\_id=19424](http://portal.opengeospatial.org/files/?artifact_id=19424), accessed 17 February 2012.
53. Open Geospatial Consortium (2011). *Implementation Statistics*, available online at <http://www.opengeospatial.org/resource/products/stats>, accessed 21 December 2011.
54. Open Geospatial Consortium (2004). *OpenGIS Web Map Server Cookbook*, Edited by K Kolodziej, Volume 1.0.2, 4 November 2004, available online at [http://portal.opengeospatial.org/files/?artifact\\_id=7769](http://portal.opengeospatial.org/files/?artifact_id=7769), accessed 17 February 2012.
55. Open Geospatial Consortium (2009). *OWS-6 Geoprocessing Workflow Architecture Engineering Report*, Edited by B Schäffer, Volume 0.3.0, 9 September 2009, available online at <http://www.opengeospatial.org/standards/per>, accessed 19 March 2012.
56. OpenGeo. (2009). *What is GeoServer*, available online at <http://geoserver.org/display/GEOS/What+is+GeoServer>, accessed 21 June 2011.
57. Parr DM (2000). *GIS Glossary of Terms*, Park Ridge: URISA.
58. Peng Z and Tsou M (2003). *Internet GIS - Distributed geographic information services for the internet and wireless networks*. John Wiley and Sons: New Jersey, USA.
59. Potts S and Kopack M (2003). *Sams Teach Yourself Web Services in 24 Hours*. Sams publishers: United States of America.

60. Pumphrey M (2010). *SLD Cook Book*. 7 April 2010, available online at <http://projects.opengeo.org/suite/raw-attachment/ticket/622/sldcookbook.pdf>, accessed on 28 February 2011.
61. Rajabifard A, Feeney M and Williamson IP (2006). Directions for the Future of SDI development, *International Journal of Applied Earth Observation and Geoinformation*, Volume 4, pp. 11-22.
62. Rautenbach V, Coetzee S, and Iwaniak A (2012a). Orchestrating OGC web services to produce thematic maps in a spatial information infrastructure. *Computers Environment and Urban Systems*, available online at <http://www.sciencedirect.com/science/article/pii/S0198971512000774> accessed 5 September 2012.
63. Rautenbach V, Coetzee S, Strzelecki M and Iwaniak A (2012b). Results of an evaluation of the orchestration capabilities of the ZOO project and the 52° North framework for an intelligent geoportal, *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Volume I-4, 2012 XXII ISPRS Congress, 25 August – 01 September 2012, Melbourne, Australia.
64. Rita E, Borbinha J and Martins B (2010). Extending SLD and SE for Cartograms. *FOSS4G 2010*, 6-9 September 2010, Barcelona, Spain, available online at <http://www.gsd.org/gsdiconf/gsd12/papers/54.pdf>, accessed on 15 February 2011.
65. Robinson AH, Morrison JL Muehrcke, Jon Kimerling A and Guptill SC (1995). *Elements of Cartography*, 6th Edition, John Wiley & Sons: United States of America.
66. Sae-Tang A and Ertz O (2007). Towards Web Service Dedicated to Thematic Mapping, *OSGeo Journal*, Volume 3, pp. 31-34.
67. Sandvik B (2008). *Using KML for Thematic Mapping*, MSc Thesis, University of Edinburgh, Edinburgh.
68. Sarang P, Jennings F, Juric M and Loganathan R (2007). *SOA Approach to Integration: XML, Web Services, ESB, and BPEL in Real-World SOA Projects*, Packt Publishing: Birmingham.
69. Schaeffer (2008) Towards a Transactional Web Processing Service (WPS-T), *GI Days 2008*, Muenster, Germany, 16-17 June 2008.
70. Slocum T, McMaster R, Kessler F and Howard H (2009). *Thematic Cartography and Geovisualization*, 3rd Edition, Prentice Hall: Upper Saddle River.
71. Stern B, Ysakowski Y and Hurni L (2006). *Statistics for Thematic Cartography*, available online at <http://www.gitta.info/Statistics/en/text/Statistics.pdf>, accessed on 21 June 2011.

72. Stock K, Stojanovic T, Reitsma F, Ou Y, Bishr M, Ortmann J and Robertson A (2012) To ontologies or not to ontologies: An information model for a geospatial knowledge infrastructure, *Computers and Geoscience*, Volume 45, pp. 98-108.
73. Stollberg B and Zipf A (2007). *OGC Web Processing Service Interface for Web Service Orchestration*, In *Lecture Notes in Computer Science*, Volume 4857, pp. 239- 251, Springer: Berlin and Heidelberg.
74. Sommerville I (2010) *Software Engineering*, 9<sup>th</sup> Edition, Addison-Wesley.
75. SourceForge (2011). *GeoServer: Download statistics*, available online at [http://sourceforge.net/project/stats/detail.php?group\\_id=25086&ugn=geoserver&mode=year&&type=prdownload](http://sourceforge.net/project/stats/detail.php?group_id=25086&ugn=geoserver&mode=year&&type=prdownload), accessed 9 August 2011.
76. Spatial Data Infrastructure Act, Act No 54 of 2003, Cape Town, South Africa.
77. StatsSA (2001). *South African Census data*, South Africa.
78. Suazo NC and Aguirre JOO (2005). Aspect-oriented web services orchestration, *2nd International Conference on Electrical and Electronics Engineering and XI Conference on Electrical Engineering*, 7-9 September 2005, Mexico City, Mexico.
79. Sui DZ and Holt JB (2008). Visualizing and Analysing Public-Health Data Using Value-by-Area Cartograms: Towards a New Synthetic Framework, *Cartographica: The International Journal for Geographic Information and Geovisualization*, Volume 43, pp. 3-20.
80. Sun J, Liu Y, Dong JS, Pu G and Tan TH (2010). Model-based methods for linking web service choreography and orchestration, *Asia Pacific Software Engineering Conference*, 30 November – 3 December 2010, Sydney, Australia.
81. Szyperski C (2002). *Component Software: Beyond Object-oriented Programming*, 2nd Edition, Harlow, UK: Addison-Wesley.
82. Tait MG (2005). Implementing geoportals: applications of distributed GIS, *Computers, Environment and Urban Systems*, Volume 29, pp. 33–47.
83. The Library of Congress (2003). *Common Query Language*, 13 February 2003, available at <http://www.loc.gov/standards/sru/sru1-1archive/cql/index.html>, accessed on 3 July 2011.
84. Tsui F and Karam O (2007). *Essentials of software engineering*, Jones and Bartlett publishers: Sudbury, Massachusetts.
85. Tyner JA (2010). *Principles of Map Design*, The Guilford Press: New York.

86. Van Loenen B, Besemer JWJ and Zevenbergen JA (2009). *SDI Convergence: Research, emerging trends, and critical assessments*, Optima Grafische Communicatie: Delft, The Netherlands, 2009, available online at <http://www.ncg.knaw.nl/Publicaties/Groen/48VanLoenen.html>, accessed 16 February 2012.
87. Wade T and Sommer S (2006). *A to Z GIS*, ESRI Press: Redlands, California.
88. Weerawarana S, Curbera F, Leymann F, Storey T and Ferguson DF (2005). *Web Service Platform Architecture*, Pearson Education: Indiana.
89. Weiser A and Zipf A (2007) *Web service orchestration of OGC web services*, In Lecture notes in Geoinformation and Cartography, by Li J, Zlatanova S and Fabbri AG, pp. 239 – 254, Berlin Heidelberg, Springer, 2007.
90. World Bank (2011). *World Bank SDI Report*, Edited by MJ Jackson and Z Gardner, October 2011, available online at [http://lgosmgb2.nottingham.ac.uk/elogeowiki/index.php/World\\_Bank\\_SDI\\_Report](http://lgosmgb2.nottingham.ac.uk/elogeowiki/index.php/World_Bank_SDI_Report), accessed 18 February 2012.
91. Woźniak E, Iwaniak A and Netzeł P (2011). Wykorzystanie standardów OGC do wizualizacji danych w infrastrukturze informacji przestrzennej [Use of OGC standards for data visualization the spatial information infrastructure], *Przegląd Geodezyjny*, Wrocław, Poland, May 2011. (In Polish).
92. Yi S, Li Q and Cheng J (1999). An Interoperability GIS Model Based on the Spatial Information Infrastructure, *Geoinformatics '99 Conference*, University of California, Berkeley, pp. 1-5.
93. Yue P, Di L, Yang W, Yu G and Zhao P (2007). Semantics-based automatic composition of geospatial Web service chains, *Computers & Geosciences*, Volume 33, pp. 649-665.
94. Zaharia R, Vasiliu L, Hoffman J and Klien E (2008) Semantic Execution meets Geospatial Web Services: A Pilot Application, *Transactions in GIS*, Volume 12, pp. 59-73.
95. Zeng L, Benatallah B, Dumas M, Kalagnanam J and Sheng Q (2003). Quality driven web services composition, *12th International Conference on World Wide Web*, pp. 411-421, 20-24 May 2003, Budapest.

## Referenced standards

1. ISO/IEC 2382-1: 1993 (1993). *Information Technology - Vocabulary*, International Organization for Standards (ISO), Geneva, Switzerland.
2. ISO 19119: 2005 (2005). *Geographic information - Services*, International Organization for Standardisation (ISO), Geneva, Switzerland.
3. OASIS (2003). *Business Process Execution Language for Web Services 1.1*, Organization for the Advancement of Structured Information Standards, available online at <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>, accessed 17 February 2012.
4. OMG (2011). *Business Process Model and Notation 2.0*, Object Management Group, available online at <http://www.omg.org/spec/BPMN/2.0/>, accessed 17 February 2012.
5. Open Geospatial Consortium (2002). *OpenGIS Web Map Server Implementation Specification 1.1.1*, available online at <http://www.opengeospatial.org/standards/wms>, accessed 28 February 2011.
6. Open Geospatial Consortium (2005). *Feature Portrayal Service 0.0.30*, available online at [https://portal.opengeospatial.org/modules/admin/license\\_agreement.php?suppressHeaders=0&access\\_license\\_id=3&target=http://portal.opengeospatial.org/files/%3fartifact\\_id=13186](https://portal.opengeospatial.org/modules/admin/license_agreement.php?suppressHeaders=0&access_license_id=3&target=http://portal.opengeospatial.org/files/%3fartifact_id=13186), accessed 14 March 2012.
7. Open Geospatial Consortium (2006a). *OpenGIS Web Map Service Implementation Specification 1.3.0*, available online at <http://www.opengeospatial.org/standards/wms>, accessed 28 February 2011.
8. Open Geospatial Consortium (2006b). *Symbology Encoding Implementation Specification 1.1.0*, available online at <http://www.opengeospatial.org/standards/symbol>, accessed 14 March 2012.
9. Open Geospatial Consortium (2007a). *OpenGIS Catalogue Services Specification 2.0.2*, available online at <http://www.opengeospatial.org/standards/wps>, accessed 12 March 2012.
10. Open Geospatial Consortium (2007b). *OpenGIS Web Processing Service Implementation Specification 1.0.0*, available online at <http://www.opengeospatial.org/standards/wps>, accessed 12 June 2011.
11. Open Geospatial Consortium (2007c). *Geography Markup Language (GML) Encoding Standard*, available online at <http://www.opengeospatial.org/standards/gml>, accessed 2 March 2012.
12. Open Geospatial Consortium (2007d). *Styled Layer Descriptor profile of the Web Map Service Implementation Specification*, available online at <http://www.opengeospatial.org/standards/sld>, accessed 28 February 2011.

13. Open Geospatial Consortium (2010a). *OpenGIS Filter Encoding 2.0 Encoding Standard*, available online at <http://www.opengeospatial.org/standards/filter>, accessed 14 March 2012.
14. Open Geospatial Consortium (2010b). *OpenGIS Web Feature Service 2.0 Interface Standard*, available online at <http://www.opengeospatial.org/standards/wfs>, accessed 22 April 2011.
15. W3C (2002). *Web Service Choreography Interface 1.0*, World Wide Web Consortium, available online at <http://www.w3.org/TR/wsci/>, accessed 17 February 2012.



## Referenced websites

1. 52°North web site, available online at <http://52north.org>, accessed 3 July 2011.
2. ArcGIS Resource Centre web site, available online at <http://resources.arcgis.com/>, accessed 7 July 2011.
3. ColorBrewer web site, available online at <http://colorbrewer2.org/>, accessed 25 February 2011.
4. GeoCommons web site, available online at <http://geocommons.com/>, accessed 1 March 2012.
5. GeoServer web site, available online at <http://geoserver.org>, accessed 19 May 2012.
6. Geo-standards Wiki, available online at <http://geostandards.geonovum.nl/>, accessed 3 February 2012.
7. Google Charts web site, available online at <https://developers.google.com/chart/>, accessed 15 July 2012.
8. indiemapper web site, available online at <http://indiemapper.com/>, accessed 29 February 2012.
9. ISO/TC 211 web site, available online at <http://www.isotc211.org/>, accessed 25 February 2012.
10. OGC implementing product details GeoServer, available online at <http://www.opengeospatial.org/resource/products/details/?pid=1041>, accessed 6 May 2012.
11. OpenLayers web site, available online at <http://openlayers.org>, accessed 8 August 2011.
12. Quantum GIS web site, available online at <http://www.qgis.org/>, accessed 3 July 2011.
13. The Thematic API web site, available online at <http://thematicmapping.org/>, accessed 3 July 2011.
14. TypeBrewer web site, available online at <http://www.typebrewer.org/>, accessed 24 February 2012.
15. Vischeck web site, available online at [www.vischeck.com/vischeck](http://www.vischeck.com/vischeck), accessed 15 February 2012.
16. ZOO Project web site, available online at <http://www.zoo-project.org/>, accessed 2 November 2011.

## Appendix A. Acronyms and abbreviations

**Table 12. List of abbreviations used in this dissertation**

API	Application Programming Interface
CSS	Cascading Style Sheets
CQL	Common Query Language
FE	Filter Encoding
FTP	File Transfer Protocol
GDAS	Geolinked Data Access Service
GIF	Graphics Interchange Format
GIS	Geographic Information System
GLS	Geographic Linkage Service
GML	Geography Markup Language
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment
INSPIRE	Infrastructure for Spatial Information in the European Community
ISO	International Standards Organization
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation
KML	Keyhole Mark-up Language
OASIS	Organization for the Advancement of Structured Information Standards
OGC	Open Geospatial Consortium
OWS	OGC Web Service
PNG	Portable Network Graphics
QGIS	Quantum GIS
REST	Representational State Transfer
SDI	Spatial Data Infrastructure
SDSS	Spatial Decision Support System
SE	Symbol Encoding
SII	Spatial Information Infrastructure
SKI	Spatial Knowledge Infrastructure
SLD	Style Layer Descriptor
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SQL	Sequence Query Language
SRS	Spatial Reference System
SVG	Scalable Vector Graphics

TSE	Thematic Symbology Encoding
UDDI	Universal Description and Discovery Interface
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WfMC	Workflow Management Coalition
WFS	Web feature service
WMS	Web Map Service
WPS	Web Processing Service
WSCI	Web Service Choreography Interface
WSDL	Web Service Description Language
WSFL	Web Service Flow Language
WWW	World Wide Web
XML	Extensible Markup Language
XPDL	XML Process Definition Language
XSLT	Extensible Style sheet Language Transformations

## Appendix B. OGC web service standards request parameters

**Table 13. The parameters of a GetFeature request (OGC 2010b)**

URL Component	Mandatory/Optional	Description
REQUEST=GetFeature	M	The name of the WFS request.
OUTPUTFORMAT	O	The default must be supported, text/xml; subtype=gml/3.1.1. Other output formats can be added, but must be advertised in the GetCapabilities.
RESULTTYPE	O	The resulttype parameter is used to indicate whether WFS should generate a complete response document or whether it should generate an empty response document indicating only the number of features that the query would return.
PROPERTYNAME	O	A list of properties may be specified for each feature type that is being queried.
FEATUREVERSION=[ALL   N]	O	If versioning is supported, the FEATUREVERSION parameter directs the WFS on which feature version to fetch.
MAXFEATURES=N	O	The maximum number of features that the WFS should return in response to a query.
SRSNAME	O	This parameter is used to specify a WFS-supported SRS that should be used for returned feature geometries.
TYPENAME (Optional if FEATUREID is specified.)	M	A list of feature type names to query.

**Table 14. Parts of Execute operation request**

<b>Name</b>	<b>Definition</b>	<b>Data type and value</b>	<b>Multiplicity and use</b>
service	Service type identifier	Character String type, not empty Value is OWS type abbreviation, namely "WPS"	One (mandatory)
request	Operation name	Character String type, not empty Value is operation name, namely "Execute"	One (mandatory)
version	Specification version for operation	Character String type, not empty Value is specified by each Implementation Specification and Schemas version	One (mandatory)
Identifier	Unambiguous identifier or name of a process	ows:CodeType, as adaptation of MD_Identifier in ISO 19115 Value is process Identifier used in Capabilities document.	One (mandatory)
DataInputs	List of inputs provided to this process execution	DataInputs data structure	Zero or one (optional) Include if any input
Response Form	Defines the response type of the WPS, either raw data or XML document. If absent, the response shall be a response document, which includes all outputs, encoded in the response.	ResponseForm type data structure	Zero or one (optional) Include when rawDataOutput or non-default outputs are requested
language	Language identifier	Character string type, RFC4646 language code of the human readable text. Must be a language listed in the Capabilities Languages element.	One (mandatory)

**Table 15. The parameters of a GetMap request (OGC 2003)**

Request parameter	Mandatory/Optional	Description
VERSION=1.3.0	M	Request version.
REQUEST=GetMap	M	Request name.
LAYERS=layer_list	M	Comma-separated list of one or more map layers.
STYLES=style_list	M	Comma-separated list of one rendering style per requested layer.
CRS=namespace:identifier	M	Coordinate reference system.
BBOX=minx,miny,maxx,maxy	M	Bounding box corners (lower left, upper right) in CRS units.
WIDTH=output_width	M	Width in pixels of map picture.
HEIGHT=output_height	M	Height in pixels of map picture.
FORMAT=output_format	M	Output format of map.
TRANSPARENT=TRUE FALSE	O	Background transparency of map (default=FALSE).
BGCOLOR=color_value	O	Hexadecimal red-green-blue colour value for the background color (default=0xFFFFFF).
EXCEPTIONS=exception_format	O	The format in which exceptions are to be reported by the WMS (default=XML).
TIME=time	O	Time value of layer desired.
ELEVATION=elevation	O	Elevation of layer desired.
Other sample dimension(s)	O	Value of other dimensions as appropriate.

## Appendix C. Examples of thematic maps produced by the ThematicWS service

The appendix provides examples of requests and the resulting thematic maps when querying the GetThematicMap function.

```
http://localhost:9090/thematic.aspx?
```

```
REQUEST=GetThematicMap&  
VERSION=1.3.0&  
LAYERS=topp%3Astates&  
STYLES=&  
FORMAT=image%2Fpng&  
CRS=EPSG%3A4326&  
BBOX=-139.84870868359,18.549281576172,-51.852562316406,55.778420423828&  
WIDTH=780&  
HEIGHT=330&  
ATTRIBUTE=population&  
THEMATICTYPE=choropleth&  
CLASED=TRUE&  
CLASSIFICATION=equal_interval&  
NROFCLASSES=6
```

Figure 65. Request for producing an equal interval choropleth thematic map with 6 classes

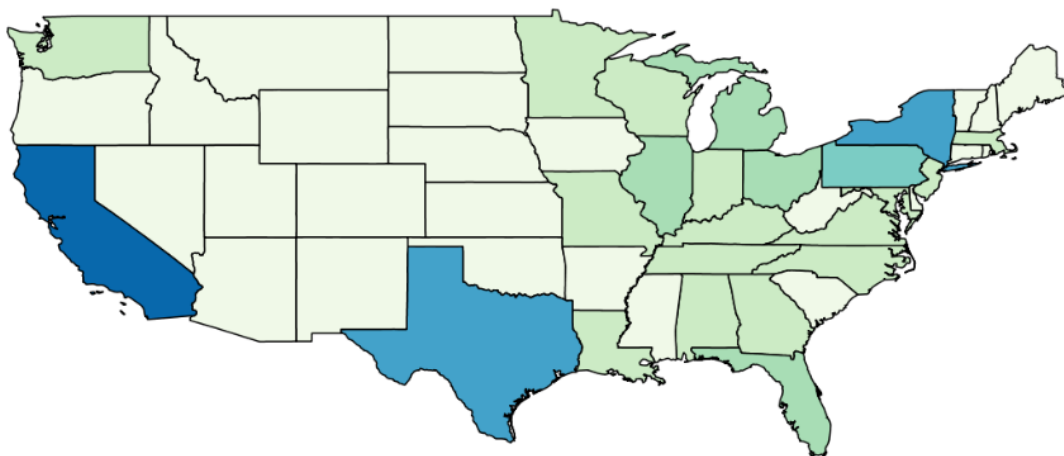
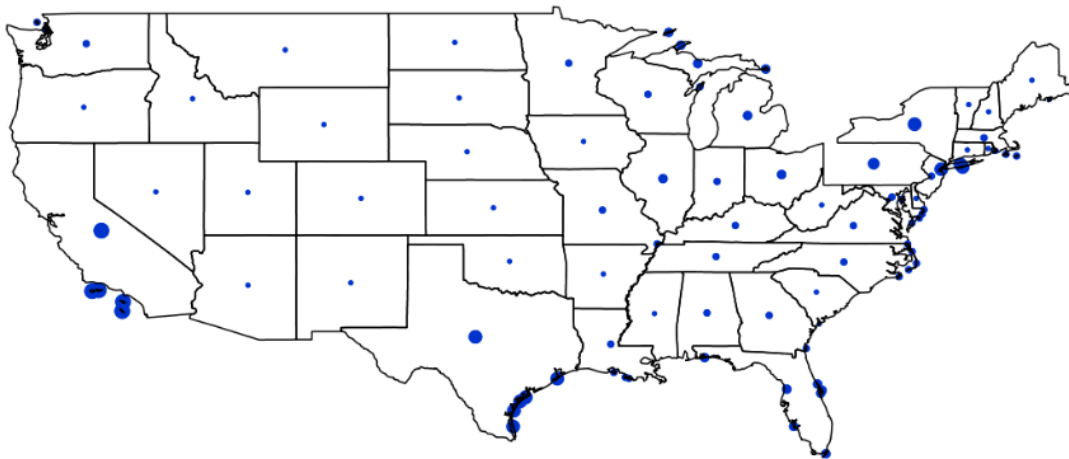


Figure 66. A classed choropleth map (equal interval classification with 6 classes) of the population of the United States of America

```
http://localhost:9090/thematic.aspx?  
REQUEST=GetThematicMap&  
VERSION=1.3.0&  
LAYERS=topp%3Astates&  
STYLES=&  
FORMAT=image%2Fpng&  
CRS=EPSG%3A4326&  
BBOX=-139.84870868359,18.549281576172,-51.852562316406,55.778420423828&  
WIDTH=780&  
HEIGHT=330&  
ATTRIBUTE=population&  
THEMATICTYPE=prop_symbol&  
CLASED=TRUE&  
CLASSIFICATION=equal_interval&  
NROFCLASSES=6
```

**Figure 67. Request for producing an equal interval proportional symbol thematic map with 6 classes**

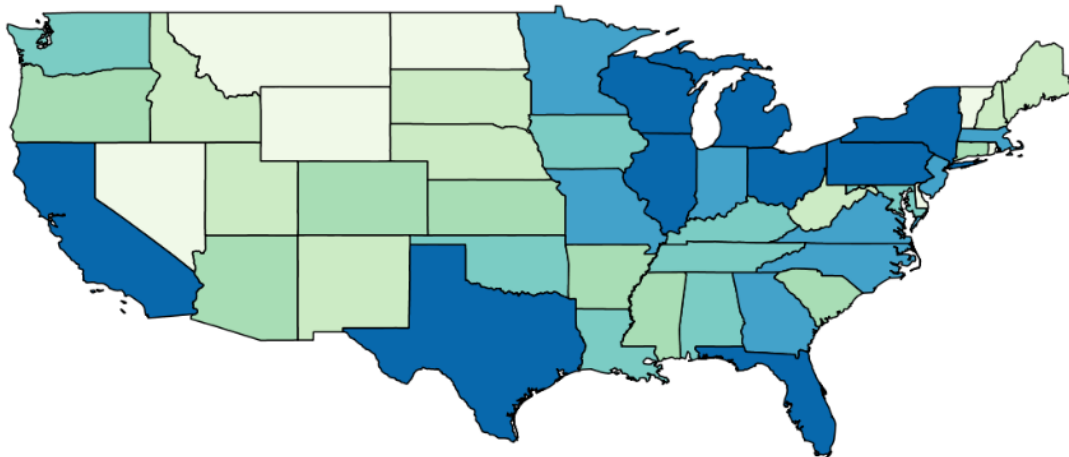


**Figure 68. A classed proportional symbol map (equal interval classification with 6 classes) of the population of the United States of America**



```
http://localhost:9090/thematic.aspx?  
REQUEST=GetThematicMap&  
VERSION=1.3.0&  
LAYERS=topp%3Astates&  
STYLES=&  
FORMAT=image%2Fpng&  
CRS=EPSG%3A4326&  
BBOX=-139.84870868359,18.549281576172,-51.852562316406,55.778420423828&  
WIDTH=780&  
HEIGHT=330&  
ATTRIBUTE=population&  
THEMATICTYPE=choropleth&  
CLASSED=TRUE&  
CLASSIFICATION=quantile&  
NROFCLASSES=6
```

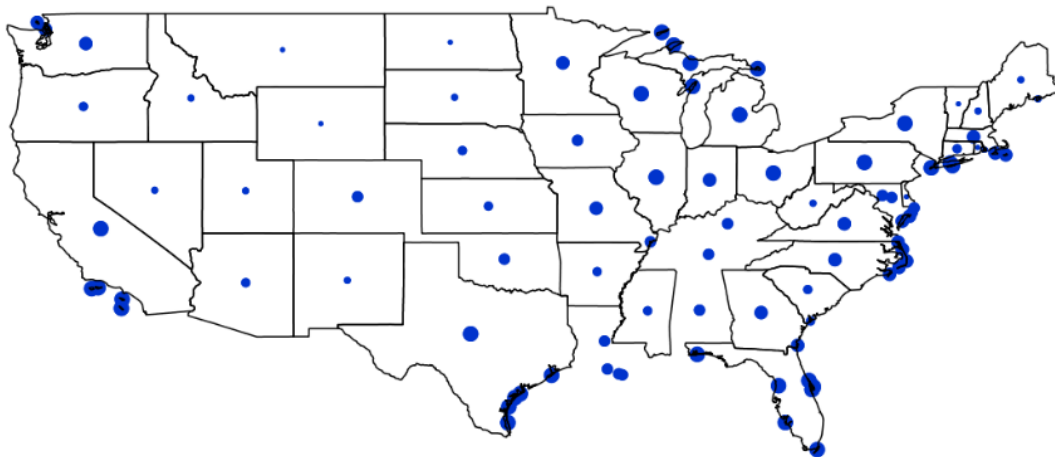
**Figure 69. Request for producing a quantile interval choropleth thematic map with 6 classes**



**Figure 70. A classed choropleth map (quantile classification with 6 classes) of the population of the United States of America**

```
http://localhost:9090/thematic.aspx?  
REQUEST=GetThematicMap&  
VERSION=1.3.0&  
LAYERS=topp%3Astates&  
STYLES=&  
FORMAT=image%2Fpng&  
CRS=EPSG%3A4326&  
BBOX=-139.84870868359,18.549281576172,-51.852562316406,55.778420423828&  
WIDTH=780&  
HEIGHT=330&  
ATTRIBUTE=population&  
THEMATICTYPE=prop_symbol&  
CLASED=TRUE&  
CLASSIFICATION=quantile&  
NROFCLASSES=6
```

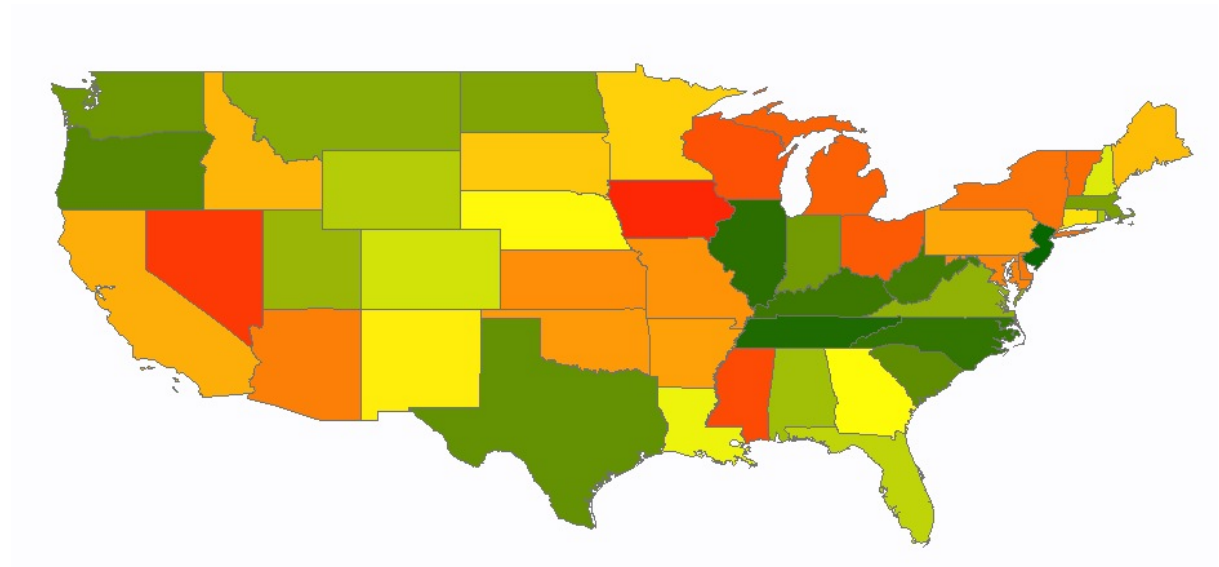
**Figure 71. Request for producing a quantile interval proportional symbol thematic map with 6 classes**



**Figure 72. A classed proportional symbol map (quantile classification with 6 classes) of the population of the United States of America**

```
http://localhost:9090/thematic.aspx?  
REQUEST=GetThematicMap&  
VERSION=1.3.0&  
LAYERS=topp%3Astates&  
STYLES=&  
FORMAT=image%2Fpng&  
CRS=EPSG%3A4326&  
BBOX=-139.84870868359,18.549281576172,-51.852562316406,55.778420423828&  
WIDTH=780&  
HEIGHT=330&  
ATTRIBUTE=population&  
THEMATICTYPE=choropleth&  
CLASED=FALSE
```

**Figure 73. Request for producing an unclassed choropleth thematic map**



**Figure 74. An unclassed choropleth map of the population of the United States of America**

```
http://localhost:9090/thematic.aspx?  
REQUEST=GetThematicMap&  
VERSION=1.3.0&  
LAYERS=topp%3Astates&  
STYLES=&  
FORMAT=image%2Fpng&  
CRS=EPSG%3A4326&  
BBOX=-139.84870868359,18.549281576172,-51.852562316406,55.778420423828&  
WIDTH=780&  
HEIGHT=330&  
ATTRIBUTE=population&  
THEMATICTYPE=prop_symbol&  
CLASED=FALSE&  
SYMBOL=SQUARE&  
SYMBOLCOLOR=FF0000
```

Figure 75. Request for producing an unclassed choropleth thematic map

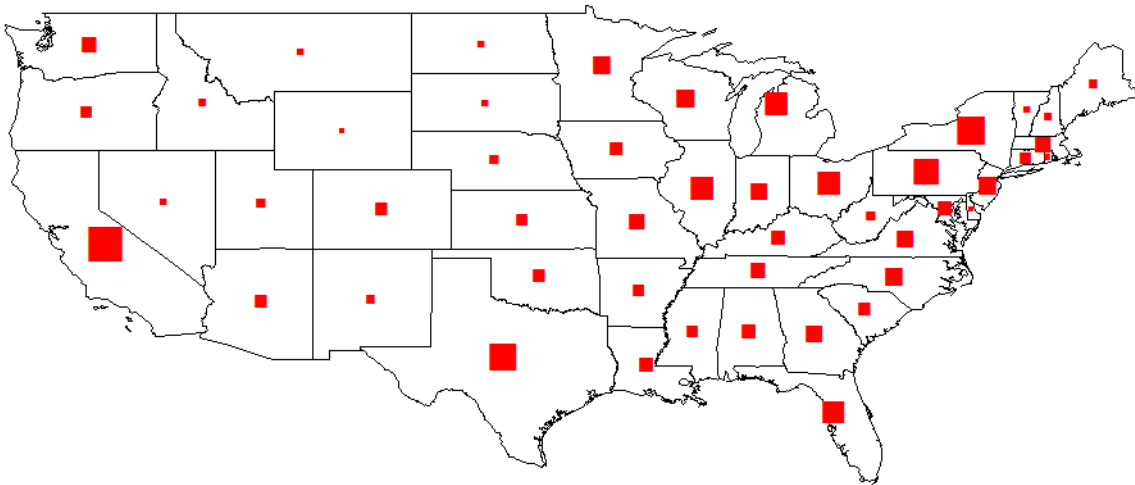


Figure 76. An unclassed proportional symbol map of the population of the United States of America

## Appendix D. ThematicWS functions

### D.1 Statistical processing service functions

#### 1. getAvg function

```
public int getAvg (ArrayList<Integer> values)
```

Calculates the average value of a specified set.

**Parameters:**

`values` – an ArrayList containing a set of integer values

**Return:**

the average value of the specified set

This function was implemented as follows: a for-loop was used to iterate through the ArrayList adding all the values in the set, the result of this was then divided by the size of the ArrayList and returned as an integer.

#### 2. getSum function

```
public int getSum (ArrayList<Integer> values)
```

Calculates the sum value of a specified set.

**Parameters:**

`values` – an ArrayList containing a set of integer values

**Return:**

the sum value of the specified set

This function was implemented as follows: a for-loop was used to iterate through the ArrayList adding all the values in the set, the result was returned as an integer.

#### 3. getCount function

```
public int getCount (ArrayList<Integer> values)
```

Return the number of values contained in a specified set.

**Parameters:**

`values` – an ArrayList containing a set of integer values

**Return:**

return the number of values

This function was implemented as follows: the size function was called on the ArrayList, which returns the size or number of items in the ArrayList, this is equal to the count, and thus returned as an integer.

#### 4. **getStdDev function**

```
public int getStdDev (ArrayList<Integer> values)
```

Calculates the standard deviation of a specified set.

**Parameters:**

`values` – an ArrayList containing a set of integer values

**Return:**

the standard deviation of the specified set

This function was implemented as follows: to calculate the standard deviation the average and deviance are firstly required. The average was obtained by calling the `getAvg` function, and iterating through the list and subtracting the average from each value and saving the new value in the list again calculated the deviance. A second iteration was required to get the square of each item. The list with these values was then sent as the parameter when the `getSum` function was called. The square root of the sum divided the size of the ArrayList produced the standard deviation.

#### 5. **getMinimum function**

```
public int getMinimum (ArrayList<Integer> values)
```

Calculates the minimum value of a specified set.

**Parameters:**

`values` – an ArrayList containing a set of integer values

**Return:**

the minimum value of the specified set

This function was implemented as follows: the first value in the list was saved as the *smallest* value, then a for-loop was used to iterate through the ArrayList comparing each value to the *smallest* value, if the value is smaller this value becomes the new *smallest* value. When the end of the ArrayList is reached, the value in *smallest* is return as an integer.

## 6. **getMaximum function**

```
public int getMaximum (ArrayList<Integer> values)
```

Calculates the maximum value of a specified set.

### **Parameters:**

`values` – an ArrayList containing a set of integer values

### **Return:**

the maximum value of the specified set

This function was implemented as follows: the first value in the list was saved as the *largest* value, then a for-loop was used to iterate through the ArrayList comparing each value to the *largest* value, if the value is larger this value becomes the new largest value. When the end of the ArrayList is reached, the value in *largest* is return as an integer.

## 7. **equalInterval function**

```
public ArrayList<Integer> equalInterval (ArrayList<Integer> values, int  
nr_of_classes)
```

Classifies the specified set of values into classes using the equal interval classification method

### **Parameters:**

`values` – an ArrayList containing a set of integer values

`nr_of_classes` – the number of classes the set should be divided into

### **Return:**

a set of integer values representing the class limits

The theory of the equal interval classification method is explained in detail in Section 2.5.1. These steps were easily ported to a function.

## 8. **quantileClass** function

```
public ArrayList<Integer> quantileClass (ArrayList<Integer> values, int  
nr_of_classes)
```

Classifies the specified set of values into classes using the quantile classification method

### **Parameters:**

`values` – an ArrayList containing a set of integer values

`nr_of_classes` – the number of classes the set should be divided into

### **Return:**

a set of integer values representing the class limits

The theory of the equal interval classification method is explained in detail in Section 2.5.2.

## 9. **stdDevClass** function

```
public ArrayList<Integer> stdDevClass (ArrayList<Integer> values, int  
nr_of_classes)
```

Classifies the specified set of values into classes using the standard deviation classification method

### **Parameters:**

`values` – an ArrayList containing a set of integer values

`nr_of_classes` – the number of classes the set should be divided into

### **Return:**

a set of integer values representing the class limits

The theory of the equal interval classification method is explained in detail in Section 2.5.3.

## **D.2 SLD style sheet preparation service functions**

### 1. **getSLD** function

```
public String getSLD (String thematictype, ArrayList<Integer>  
class_limits, int nr_of_classes, String request_parameter)
```

Generates the SLD according to the specifications



**Parameters:**

`thematictype` – specifies the type of thematic map, can be either `choropleth`/`prop_symbol`

`class_limits` – an `ArrayList` containing a set of integer values

`nr_of_classes` – the number of classes the set should be divided into

`request_parameter` – the attribute value according to which the data was classified

**Return:**

the XML SLD style sheet in string format

The `getSLD` function builds-up the SLD style sheet according to the class limits calculated and the type of thematic map requested. The function firstly uses an if-statement to determine if a choropleth or proportional symbol map should be produced. Using a for-loop the different rules of the style sheet is created. The difference between the choropleth and proportional symbol map is the type of symbolizer used within the rules, for a choropleth map a `PolygonSymbolizer` is used and for a proportional symbol map the `PointSymbolizer` is used. A choropleth map also requires a hue of colours that is assigned to each class; the colour hue is obtained by calling the `getColors` function.

**2. getColors**

```
private ArrayList<String> getColors (int nr_of_classes)
```

Returns a set of colours

**Parameters:**

`nr_of_classes` – the number of classes the set should be divided into

**Return:**

a set of hex colours codes

The `getColors` function has a predefined set of colours that are seen as optimal by the ColorBrewer website. The only parameter of the function is the number of classes, or colours in this instance. A set of colours that are optimal for the specified size is then returned. The hex colour codes are return in an `ArrayList`.

## **Appendix E. Journal publication and referenced conference paper**

### **E.1 Results of an evaluation of the orchestration capabilities of the ZOO project and the 52° North framework for an intelligent geoportal**

**Rautenbach V**, Coetzee S, Strzelecki M and Iwaniak A (2012). Results of an evaluation of the orchestration capabilities of the ZOO project and the 52° North framework for an intelligent geoportal, *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Volume I-4, 2012 XXII ISPRS Congress, 25 August – 01 September 2012, Melbourne, Australia.

The paper is included as Chapter 8 of the dissertation with some minor changes.

## E.2 Orchestrating OGC web services to produce thematic maps in a spatial information infrastructure

**Rautenbach V**, Coetzee S, and Iwaniak A (2013). Orchestrating OGC web services to produce thematic maps in a spatial information infrastructure, *Computers Environment and Urban Systems*, ISSN 0198-9715, 37:107 - 120, DOI: [10.1016/j.compenvurbsys.2012.08.001](https://doi.org/10.1016/j.compenvurbsys.2012.08.001).