Data Enrichment for Adaptive Map Generalization Using Web Services

Dissertation

zur
Erlangung der naturwissenschaftlichen Doktorwürde
(Dr. sc. nat.)

vorgelegt der
Mathematisch-naturwissenschaftlichen Fakultät
der
Universität Zürich

von

Moritz Neun

aus Deutschland

Promotionskomitee
Prof. Dr. Robert Weibel (Vorsitz)
Dr. Dirk Burghardt

Zürich, 2007

Summary

Map generalization seeks to maintain and improve the legibility on a reduced map scale with less space for displaying the desired map content. Therefore unimportant detail has to be simplified or omitted while the salient properties and the structural characteristics have to be preserved or even highlighted. In a generalization process several cartographic tasks are applied such as the selection of important map objects, the shape simplification or exaggeration, the aggregation or the displacement of map objects. Thereby a broad range of cartographic principles have to be respected simultaneously while applying the tasks in the right configuration. For the automation of generalization this holistic cognitive reasoning process, usually performed by skilled human cartographers, has to be accomplished by a computer. This requires software algorithms for performing the cartographic tasks and techniques to control and evaluate the application of the algorithms in the whole generalization process.

This thesis investigates the provision of auxiliary data, also termed *data enrichment*, and its use for controlling an automated adaptive generalization process. The term *adaptive generalization* expresses the desire to control the application of the generalization tasks, the so-called operators. The data enrichment provides *structural knowledge* about the spatial and semantical context of the map features in order to support this complex decision making process. Up to now the structural knowledge was only used implicitly in specific generalization algorithms but rarely made available for the use and re-use in a whole generalization process. Therefore, this thesis proposes a services-based system called WebGen which facilitates the provision of supporting structural knowledge and of generalization operators in a platform independent way for being used in an automated generalization process.

Generalization algorithms as well as data enrichment methods are developed by various research groups but usually on their computing platform. An open generalization research platform would allow to test and share the methods developed by the different researchers in order to boost future research. The technology proposed with the WebGen framework fulfills the requirements of such an open generalization research platform by allowing the platform independent exchange and use of functionalities.

The core of this thesis consists of five scientific Research Papers which treat the three main objectives: (1) Storage and provision of enriched data using generalization services, (2) Identification and formalization of structural relationships for data enrichment, (3) Exploitation of enriched data for adaptive generalization. *Research*

Paper 1 introduces the topic of generalization frameworks and provides a motivation for the development of a services based generalization platform. The requirements of such a platform are discussed and two open architectures are described and illustrated with first experiments. *Research Paper 2* describes the development of a classification and framework of generalization web services and shows as proof of concept the implementation of the WebGen framework. *Research Paper 3* focuses on the category of the support services for the provision of enriching information to generalization operators and processes. This paper provides a classification of structural knowledge expressed through relations between map objects. *Research Papers 4 and 5* show the development and experiments with process services for the automatic chaining of generalization operators. These process services provide a successful exploitation of the enriching information provided by support services. Parallelization techniques and furthermore a continuous learning knowledge base are used for increasing the process performance.

The main contribution of this thesis is the use of a services oriented architecture for an interoperable gen-eralization system, which can serve for the provision of data enrichment and also as common research platform. Thereby a conceptual framework including three service categories (support, operator, process service) is developed and the feasibility of the approach is proven with the implementation of the WebGen framework. This framework allows the loose coupling of different generalization algorithms and other supporting methods for use in complex generalization workflows. It can therefore be used as a solution for a common interoperable generalization platform to boost research on generalization. Finally this thesis highlights some future challenges in the application of the proposed methods to form a comprehensive generic fully automated generalization system.

Zusammenfassung

Bei einer Verkleinerung des Massstabes hat eine Karte weniger Platz um den gewünschten Inhalt darzustellen. Die Aufgabe der kartographischen Generalisierung ist es, dabei die Lesbarkeit der Karte zu erhalten oder zu erhöhen. Zu diesem Zweck müssen unwichtige Details weggelassen und die herausragenden, wichtigen Eigenschaften sowie der strukturelle Charakter erhalten oder sogar hervorgehoben werden. Im Prozess der Generalisierung werden verschiedene kartographische Aufgaben, wie die Selektion von wichtigen Kartenobjekten, die Vereinfachung oder Hervorhebung von Objektformen oder auch das Verschmelzen oder Verschieben von Objekten ausgeführt, die in der richtigen Anordnung und mit den richtigen Einstellungen angewandt werden müssen. Gleichzeitig muss eine grosse Anzahl kartographischer Prinzipien beachtet werden. Diese umfassende gedankliche Leistung wird normalerweise von erfahrenen Kartographen erbracht und soll bei der automatischen Generalisierung vom Computer geleistet werden. Dazu braucht es Softwarealgorithmen für die Durchführung der kartographischen Aufgaben, die so genannten Operatoren, sowie Techniken um die Ausführung der Operatoren und des ganzen Generalisierungsprozesses zu steuern und zu überwachen.

Thema dieser Arbeit ist die Datenanreicherung, also die Bereitstellung von unterstützenden Zusatzdaten, sowie deren Nutzung für die Steuerung der Anwendung der Operatoren in einem automatisierten, adaptiven Generalisierungsprozess. Die Datenanreicherung unterstützt die komplexen Entscheidungsprozesse der adaptiven Generalisierung mit strukturellem Wissen über den räumlichen und semantischen Kontext der Kartenobjekte. Diese zusätzlichen Informationen wurden bisher meist implizit direkt in den spezifischen Algorithmen erzeugt und auch benutzt. Durch die Datenanreicherung sollen sie nun stattdessen für die wiederholte Benutzung während eines ganzen Generalisierungsprozesses zur Verfügung stehen. Dazu wird in dieser Arbeit eine verteilte Service-Architektur, genannt WebGen, vorgestellt. Dieses System erlaubt es sowohl Datenanreicherungsmethoden als auch Generalisierungsalgorithmen als plattformunabhängige modulare Web-Services zur Verfügung zu stellen um sie dann für verschiedene Generalisierungsprozesse zu verwenden.

Eine Reihe von Forschern und Forschungsgruppen entwickeln Methoden für die Generalisierung und Datenanreicherung. Dabei werden oft unterschiedliche Computer- und Software-Plattformen benutzt, was den Austausch untereinander und das Testen der verschiedenen Entwicklungen sehr schwierig macht. Zukünftige Forschung würde deshalb von einer gemeinsamen, offenen Forschungsplattform profitieren. Die WebGen Plattform erfüllt die Anforderungen an eine solche offene Forschungsplatt-

form, da sie den Austausch und die Benutzung von Funktionalitäten über Plattformgrenzen hinweg erlaubt.

Der Kern dieser Arbeit besteht aus fünf Forschungspublikationen, die sich mit den drei Hauptzielen der Arbeit befassen: (1) Speicherung und Bereitstellung der Datenanreicherung als Generalisierungs-Services, (2) Identifikation und Formalisierung struktureller Eigenschaften für die Datenanreicherung, (3) Nutzung der Datenanreicherung für die adaptive Generalisierung. In *Forschungspublikation 1* wird die Motivation und Thematik einer offenen, gemeinsamen Forschungsplattform eingeführt. Die Anforderungen einer solchen Plattform werden diskutiert und zwei Ansätze anhand von ersten Experimenten illustriert. *Forschungspublikation 2* präsentiert eine Klassifikation von Web-Services für die Generalisierung und zeigt als Machbarkeitsstudie die Implementierung der WebGen Plattform. *Forschungspublikation 3* konzentriert sich auf die so genannten "support services" zur Bereitstellung von unterstützenden Daten und Methoden wie der Datenanreicherung für die Benutzung durch Generalisierungsoperatoren und zur Steuerung von ganzen Prozessen. In einer Klassifikation werden die verschiedenen Arten von strukturellem Wissen für die Datenanreicherung erfasst. Die *Forschungspublikationen 4 und 5* demonstrieren die Anwendung der Datenanreicherung für die Steuerung von Generalisierungsprozessen. Diese so genannten "process services" benutzen die Information der "support services" für die automatische Verkettung und Steuerung von verschiedenen Generalisierungsoperatoren. Für eine Steigerung der Prozessperformance werden Parallelisierung sowie selbst lernende Verfahren eingesetzt.

Der massgebliche Beitrag dieser Arbeit ist die Verwendung von Web-Service Technologien für die Schaffung eines offenen, plattformunabhängigen Generalisierungssystems, das für die Bereitstellung der Datenanreicherung und auch als offene gemeinsame Forschungsplattform genutzt werden kann. Das konzeptionelle Gerüst unterscheidet dabei drei Service-Kategorien (support, operator, process). Die Machbarkeit wurde mit der Implementierung der WebGen Plattform gezeigt. Mit dieser Plattform können verschiedene Generalisierungsalgorithmen und unterstützende Methoden wie die Datenanreicherung zusammen in komplexen Generalisierungsprozessen verwendet werden. Abschliessend gibt die Arbeit einen Ausblick für die Generalisierungsforschung im Zusammenhang mit den vorgestellten Methoden und beschreibt die nächsten Herausforderungen auf dem Weg zu einem generischen, voll automatisierten, Generalisierungssystem.

Acknowledgements

This work was carried out during my time at the Department of Geography of the University of Zürich. I want to express my thanks and acknowledgements to all the people that in many ways contributed to this thesis. Here I will name some of them and I inwardly thank the others.

I would like to thank Robert Weibel for being my chief supervisor and for giving the opportunity to carry out this research in the GIS group. I am deeply indebted to him for his support, his readings and rereadings of my paper and thesis drafts and his valuable comments and suggestions.

Dirk Burghardt acted as my supervisor and I owe him a great deal for his very valuable support and the research we could do together. He always motivated and inspired me and our fruitful collaboration can be seen in the research papers presented in this thesis.

Thanks to Stefan Steiniger for being my PhD-mate on project DEGEN. Many thanks to the other "generalizers" Alistair Edwardes, Ingo Petzold, Matthias Bobzien and Patrick Lüscher for many interesting and stimulating discussions. Thanks to Ross Purves and Sara Fabrikant for doing the promotion seminar and teaching me how to do a PhD. Thanks to Elisabeth Cottier for helping me with all the administrative stuff.

Thanks to all the colleagues from the GIS and GIVA division for the always good and friendly atmosphere. Thanks for the good time, for many discussions not only about work, for all the activities other than work and for just being friends.

I am grateful to Peter van Oosterom and Nicolas Regnauld for reading and reviewing this thesis as external experts.

I would like to thank my brother Philipp for proofreading this thesis and I owe a huge debt to my parents and the all of my family for always supporting and encouraging me. Finally I want to thank my darling Claudia Wandke for her lovely help, her patience and understanding during the time of my dissertation.

Moritz Neun, October 2007

This research work is part of the project "DEGEN: Data Enrichment for Adaptive Map Generalization" supported by the Swiss National Science Foundation (grant 20-101798).

Contents

I Synopsis **1**

1 Introduction **3**
1.1 Motivation . . . 3
1.1.1 Data Enrichment . . . 3
1.1.2 Services-Based Map Generalization . . . 4
1.1.3 Controling Adaptive Generalization Prozesses . . . 5
1.2 Thesis Rationale . . . 5
1.2.1 Objectives and Research Questions . . . 5
1.2.2 Included Research Papers . . . 7
1.3 Structure of the Thesis . . . 8

2 Theoretical Background and State-of-the-Art **9**
2.1 Principles of Map Generalization . . . 9
2.1.1 Definition . . . 9
2.1.2 Automated Map Generalization . . . 10
2.1.3 Modeling the Generalization Process . . . 11
2.1.4 Data Enrichment for Adaptive Generalization . . . 13
2.2 Generalization Services . . . 18
2.2.1 Interoperability and Distributed Environments . . . 18
2.2.2 Interoperability of Geographical Data and Algorithms . . . 22
2.3 Summary: Challenges for Research . . . 30

3 Results and Discussion **33**
3.1 Synthesis of the Research Papers . . . 33
3.1.1 Research Paper 1 . . . 35
3.1.2 Research Paper 2 . . . 37
3.1.3 Research Paper 3 . . . 41
3.1.4 Research Paper 4 . . . 45
3.1.5 Research Paper 5 . . . 48
3.2 General Discussion . . . 50

3.2.1 Strengths 50
3.2.2 Weaknesses and Open Problems 52

4 Conclusion and Outlook **55**
4.1 Main Contributions 56
4.2 Answering the Research Questions 59
4.3 Ongoing Challenges and Outlook 61
4.3.1 Building a Comprehensive Generalization System 61
4.3.2 Further Challenges and Improvements 63

Bibliography **67**

II Research Papers **79**

Research Paper 1 **81**

Research Paper 2 **103**

Research Paper 3 **117**

Research Paper 4 **147**

Research Paper 5 **169**

III Appendices **177**

A The WebGen Framework **179**

B Comment on OGC Web Processing Service (WPS) **185**

C Complete Publication List **189**

D Curriculum Vitae **191**

List of Figures

1.1 Research objectives of project DEGEN 6

2.1 Same area at three different map scales 9
2.2 Generalization with and without additional contextual information . 14
2.3 Horizontal and vertical relations 15
2.4 Standard Remote Procedure Call (RPC) 20
2.5 The "publish-find-bind" paradigm 21
2.6 Data delivery services architecture 23
2.7 Middleware architecture 23
2.8 GiMoDig service architecture 24
2.9 Processing services architecture 26
2.10 Processing services allow the upload of data and parameters 26
2.11 OXYGENE Geographic WebService 28
2.12 MapShaper.org Generalization Web Service 29

3.1 Categories of generalization services 38
3.2 Support service types 42
3.3 Conflict due to road symbolization 45
3.4 Flow chart of the applied iterative generalization process 49

4.1 System architecture for tailoring on-demand map products 62

A.1 WebGen client components 182
A.2 WebGen server components 183

List of Acronyms

AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
CPU	Central Processing Unit
DEGEN	Data Enrichment for Adaptive Generalization
DOM	Document Object Model
DTD	Document Type Definition
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
GIS	Geographic Information System
GISc	Geographic Information Science
GML	Geography Markup Language
ICA	International Cartographic Association
Java	Java programming language
JCS	Java Conflation Suite
JTS	Java Topology Suite
JUMP	JUMP Unified Mapping Platform
MAS	Multi Agent System
MRDB	Multi-Representation / Multi-Resolution Database
OGC	Open Geospatial Consortium
OWS	OGC Web Services
PDF	Portable Document Format
PHP	Hypertext Pre-processor
RDBMS	Relational Database Management System
RPC	Remote Procedure Call
RMI	Remote Method Invocation
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language

SVG	Scalable Vector Graphics
UDDI	Universal Description, Discovery, and Integration
URL	Universal Resource Locator
UTF-8	8-bit Unicode Transformation Format
WFS	Web Feature Server
WFS-T	Transactional Web Feature Server
WMS	Web Map Server
WPS	Web Processing Service
WSDL	Web Service Description Language
WWW	World Wide Web (or simply Internet)
XML	Extensible Markup Language
XSLT	Extensible Style Language Transformation

Part I

Synopsis

Chapter 1

Introduction

1.1 Motivation

Map generalization seeks to maintain and improve the legibility and ensure the desired contents of a map after the scale has been changed. On a smaller map scale less space is available for displaying the map content. Thus it is necessary to simplify or omit unimportant detail while maintaining or even highlighting the salient properties. Thereby the structural characteristics of the source data have to be preserved. A generalization process for obtaining such a simplified representation applies several cartographic tasks and has to comply with a variety of cartographic principles in order to retain the legibility as well as structural characteristics of the map. The cartographic tasks include for example the selection, the shape simplification or exaggeration, the aggregation and the displacement of map objects. Skilled human cartographers can apply these cartographic tasks while respecting the broad range of cartographic principles simultaneously. For automating this complex generalization process, however, this holistic cognitive reasoning process has to be performed by a computer. Therefore, for controlling and performing the cartographic transformations many different techniques are required including for example geometric algorithms and measures, advanced data structures and spatial database techniques, spatial analysis and pattern recognition as well as artificial intelligence approaches.

1.1.1 Data Enrichment

Many generalization algorithms are only suitable for specific problems. The term ***adaptive generalization*** expresses the intention to use the appropriate algorithm with the right parameters for a given generalization task. The selection of the most suitable algorithms – particularly when several algorithms are integrated into a comprehensive workflow – is a complex decision making process and depends critically on the availability of auxiliary information about the context. This ***structural knowledge***, being the knowledge of the spatial and semantical context of map features and their structural interrelationships is critical for the understanding of the role of the features represented on a map and thus for automated generalization (Armstrong, 1991; Ruas and Plazanet, 1996; Regnauld, 2005).

This supporting knowledge, however, has rarely been coded directly into the commonly available cartographic databases. Hence, algorithms for extracting structural knowledge from the raw cartographic data are required in order to "enrich" the available cartographic databases (Ruas and Plazanet, 1996). The activity of extracting from "raw" spatial data auxiliary information about spatial and semantic structure and relationships that are useful for map generalization, and integrating it into a spatial database, is called ***data enrichment***. This enriching structural knowledge[1] consists of derived attributes, geometries and of complex structures for modeling the relationships (see Research Paper 3 in §3.1) and it should support different processes during the automated generalization of maps. It concerns data characterization, conflict detection, indirectly also the algorithm selection, parameterization and chaining, and finally the evaluation of the map (cf. §2.1.4).

1.1.2 Services-Based Map Generalization

In recent years cartography has evolved in a new direction with the application of web technologies and service oriented architectures (SOA). As a result of this development maps in the web context generally are generated on-demand and on-the-fly, containing more specific and tailor-made information then traditional static maps that were designed in advance for general purpose usage. Currently the focus of the development of (geo-)services is on web services for data delivery which are used for web cartography as well as for the development of (national) spatial data infrastructures (Crompvoets, 2006). The Open Geospatial Consortium (OGC) specifies standards for these services (OGC, 2002) to implement web services for feature access (WFS) and web mapping (WMS) for cartographic presentation. This increasing use of dynamically provided maps has lead also to a growing need for automated map generalization solutions. Not surprisingly therefore, the OGC has mentioned the plan to develop specifications for "Feature Generalization Services" as part of their OGC Web Services (OWS) initiative (OGC, 2002) several years ago. However, the official definition of such specifications has not experienced much progress so far.

Generalization algorithms as well as auxiliary data structures providing enriching structural knowledge are being developed by various research groups, typically on their platform of choice. The problem, then, is that although the generalization toolbox is seemingly steadily filling up no real progress can be made in research nor in production as long as these tools exist on different platforms (Edwardes et al., 2003). Therefore the map generalization research community has started to develop an interest into the development of a common open research platform that would allow testing and sharing of generalization algorithms (Badard and Braun, 2003; Edwardes et al., 2003). This has been evidenced through discussions at the meetings of the ICA Commission on Map Generalization and Multiple Representation in Paris 2003 and Leicester 2004. ***Generalization services*** can be used as an interoperable

[1] The term "data enrichment" in the context of this thesis refers to auxiliary "structural knowledge" in the sense of Armstrong (1991) being derived from the "raw" spatial data. Knowledge engineering techniques (machine learning) for acquiring "procedural knowledge" are addressed only briefly in Research Paper 5 (see §3.1).

research platform allowing the combination of different algorithms and supporting data structures using a common interface (see Research Papers 1 and 2 in §3.1).

1.1.3 Controling Adaptive Generalization Prozesses

Various generalization operators have been developed to be applied to the different map objects during the generalization process. Examples include algorithms to simplify or smooth a road centerline as well as algorithms to displace or aggregate buildings. The goal of an adaptive generalization process is therefore to select the right operator for the currently treated map objects and to parameterize these operators according to the map situation. Further they must be applied in the right sequence in order to obtain the desired results. Finally the resulting map quality of such a process must be measured in order to evaluate and improve the operator selection and chaining.

So-called "constraints" can be used to describe the conditions that have to be met in order to make a map legible and compliant to the user needs (see also §2.1.3 and §3.1.4). These constraints are enriching the data with knowledge about map object characteristics and conflicts and are informing about whether the cartographic requirements are fulfilled or violated. The analysis of the constraint violations allows the selection and parameterization of generalization operators as well as the evaluation of the results. Combinatorial optimization techniques (Harrie and Weibel, 2007) for the selection and chaining of operators are a promising technique to use constraints for the control of the generalization process.

1.2 Thesis Rationale

Data enrichment equips "raw" spatial data with additional information about spatial and semantic properties and relations. This information can be used to achieve an adaptive generalization process. A services oriented architecture can serve as an open research platform for allowing the provision and exploitation of the enriching information.

The aim of this thesis is to contribute to the conceptual and methodological knowledge about the domain of data enrichment for generalization. The following section presents the objectives of the thesis and the research questions that were to be answered. Further it relates these objectives and questions to the Research Papers included in Part II of this thesis.

1.2.1 Objectives and Research Questions

This thesis was conducted in the context of the project DEGEN (**D**ata **E**nrichment for Adaptive **Gen**eralization)[2] having a focus on data enrichment, the modeling of

[2]In the project DEGEN two PhD students (Moritz Neun and Stefan Steiniger) were funded by the Swiss National Science Foundation SNF (grant no. 20-101798) for the duration of 3.5 years.

the enriched data and the exploitation of this enriched data in map generalization (cf. figure 1.1).

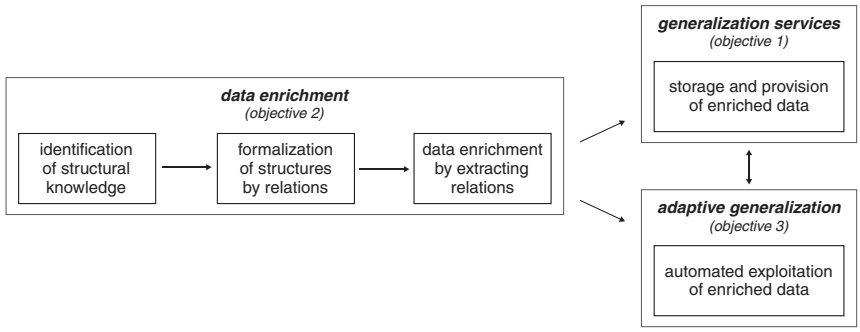


Figure 1.1: Research objectives of project DEGEN

In two sub-projects aspects of data enrichment for map generalization were treated. Each sub-project leads to a separate thesis. The sub-project leading to this thesis focused mainly on the provision and exploitation of enriched data in a generic and interoperable way. The other sub-project by Stefan Steiniger focused on the classification, modeling and extraction of enriching data especially with knowledge about relations between map objects. The identification and modeling of structural relations was the link between the two sub-projects.

The hypotheses of this thesis are that data enrichment helps generalization making adaptive generalization possible and that it is possible to build a common interoperable research platform based on generalization services which can be used for the provision and exploitation of data enrichment. Thus, the following three main objectives were treated in this thesis (see also figure 1.1):

Objective 1: Methods for the *storage and provision of enriched data* for its later exploitation by various generalization operators shall be developed. As the development of algorithms for the extraction and data models for the representation of structural knowledge is pursued by various researchers in the generalization community, a platform independent interoperable approach is desired. With the focus on *generalization services* the following research questions were to be answered:

(a) What are the requirements for an open research and development platform?
(b) Can a services based architecture be used for the sharing of algorithms and for the extraction and provision of enriching information in an interoperable way?

Objective 2: Structural knowledge, describing the spatial and semantical context of map features and their structural relationships shall be identified and formalized. For *data enrichment* the knowledge about the relations has to be extracted from the "raw" data. Thus the following research questions were to be answered:

(c) What types of structural knowledge can be provided by services for the support of generalization methods?
(d) How is the knowledge about map object relations being generated and used?
(e) Can this knowledge about relations serve as enriching information?

Objective 3: The enriching data shall be exploited for the control of an adaptive generalization process. This objective shows how the auxiliary information obtained from data enrichment can be used in order to make the adaptive generalization possible. This objective leads to the following research questions:

(f) How can an adaptive generalization process be modeled using a framework of generalization services?
(g) How can the enriched data be exploited in order to control an adaptive generalization process?

1.2.2 Included Research Papers

This thesis reports on research successively published in peer-reviewed conference proceedings and international scientific journals. The five research papers are:

Research Paper 1:

Edwardes, A., D. Burghardt, M. Neun (2007): Experiments to build an open generalisation system. In W. Mackaness, A. Ruas and T. Sarjakoski (eds), *Generalisation of geographic Information: Cartographic Modelling and Applications* chapter 8, (pp. 161-175). Amsterdam: Elsevier Science.

Research Paper 2:

Burghardt, D., M. Neun and R. Weibel (2005): Generalization Services on the Web - Classification and an Initial Prototype Implementation. *Cartography and Geographic Information Science*, Volume 32, Number 4, October 2005, pp. 257-268.

Research Paper 3:

> Neun M., D. Burghardt and R. Weibel (in press): Web Service Approaches for Providing Enriched Data Structures to Generalisation Operators. Accepted for *International Journal of Geographical Information Science*.

Research Paper 4:

> Neun M., D. Burghardt and R. Weibel (submitted): Automated Processing for Map Generalization with Modular Operator Services. Submitted to *GeoInformatica*.

Research Paper 5:

> Burghardt D. and M. Neun (2006): Automated Sequencing of Generalisation Services Based on Collaborative Filtering. In M. Raubal, H. J. Miller, A. U. Frank and M. Goodchild (eds): *Geographic Information Science*. 4th International. Conference on Geographical Information Science (GIScience 2006), IfGIprints 28, pp. 41-46.

A summary and discussion of each of these research papers can be found in chapter 3. In these five papers the objectives and research questions of this thesis are answered. Research Papers 1 and 5 are intended to provide a motivation and an outlook, respectively, whereas Papers 2, 3 and 4 treat the main objectives. A more detailed mapping between the research objectives, the research questions and the corresponding Research Papers is given in section 3.1 and section 4.1.

1.3 Structure of the Thesis

The content of this thesis is presented in two parts. In Part I (Synopsis) the above research papers are embedded in the scientific context. In Part II (Research Papers), the research papers are presented with the content and format as they were submitted or published. The ordering of these papers corresponds to the logical sequence of the research process for this thesis. The publication dates however have another ordering for the three last papers. Links in Part I referencing the Research Papers use the paper numbers above.

The Synopsis continues, after this introductory chapter, with the theoretical background and a review of the state of the art (chapter 2). First, some principles of map generalization, with respect to the focus on data enrichment, are introduced. Following this the concept of data enrichment for map generalization is introduced and relevant approaches are reviewed. The chapter is concluded by an introduction to the principles and techniques of services and interoperability in GIScience. In chapter 3 the results of the research presented in the five Research Papers are summarized and discussed. Finally the synopsis ends with a conclusion and an outlook (chapter 4).

Chapter 2

Theoretical Background and State-of-the-Art

2.1 Principles of Map Generalization

2.1.1 Definition

Maps are an abstract representation of geographical reality. Maps differ in their degree of abstraction due to differences in the map scale (Peterson, 2006) and in the map purpose. At a smaller map scale the representation of map detail must be simplified and abstracted but the essential character and also the graphical quality of the map should be retained. The International Cartographic Association defines the cartographic generalization as "*selection and simplified representation of detail appropriate to the scale and/or purpose of a map*" (ICA, 1973).

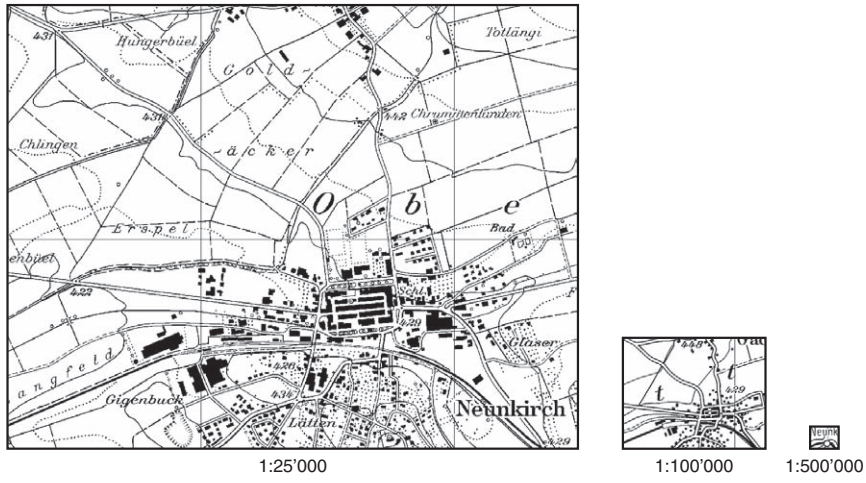


Figure 2.1: Same area at three different map scales – detail has to be reduced in order to preserve a readable map (reproduced by permission of swisstopo, BA071392)

Map generalization should emphasize the essential, suppress the unimportant and maintain logical and unambiguous relations between map objects (Weibel, 1997a). The map scale is not the only factor influencing the generalization process. Also the map purpose is important for the identification of map features to be retained, for the graphical visualization and simply for the definition of the required map scale. According to Müller (1991), generalization may be initiated by economic, data robustness, multipurpose and finally by display and communication requirements.

In the example of figure 2.1 three extracts of topographical maps at different scales (also termed levels of detail) are presented showing the same geographical area. At the coarser levels of detail the requirements of legibility, accuracy and of map content and detail must be balanced. At the 1:100'000 level of detail for example only the larger roads are maintained and small buildings have been removed or aggregated. At the scale of 1:500'000 the villages have been collapsed to a point and only the major roads are retained.

2.1.2 Automated Map Generalization

Today geographic information is used on manifold occasions. Not only paper maps but also web map services and especially geographic information systems (GIS) provoke a widespread use of geographic data. These data have to be maintained and displayed in various ways. Thus a demand for automated map generalization using GIS is evident. The goal for automated map generalization is to only once capture the geographic data at the very finest level of detail and then derive all the different desired digital and analogue products automatically.

"*Given the complexity of this task, and the absence of users with cartographic skills, it is assumed that the human should be largely absent from this process. In terms of an automated solution, what is required is an autonomous self-evaluating system able to create maps in digital or analog form that vary widely both in scale and thematic content. This is touted as the 'holy grail' of automated cartography.*" (Mackaness, 2006)

Typically various methods are used in a generalization process. For a controlled reduction of data (Morgenstern and Schürer, 1999) map objects have to be selected and reclassified. This process is usually termed *model generalization*. Other methods have to clarify the visualization. Therefore map situations have to be simplified or exaggerated and map objects are displaced or aggregated. This process is commonly termed *cartographic generalization* and also has to consider about the final symbolization of the data with resulting conflicts (Weibel, 1997b). A listing of typical generalization methods (also termed generalization operators) is given for example by McMaster and Shea (1992). For the different generalization operators various algorithms do exist. Generalization algorithms used during the experiments for this thesis are illustrated more deeply in Research Paper 4 (§3.1.4).

2.1.3 Modeling the Generalization Process

Automated generalization processes consist of several sub-processes that have to be sequenced and controlled. The modeling of an automated generalization process can be achieved with different approaches ranging from simple batch processing to sophisticated constraint based methods. Harrie and Weibel (2007) differentiate *condition-action*, *human interaction* and *constraint based* modeling as the three prevailing generalization process models apart from simple static batch processes.

The *condition-action* model (also called rule-based model) uses knowledge about characteristic structures (Brassel and Weibel, 1988) or cartometric measures (McMaster and Shea, 1992) as conditions for the triggering of actions. Rules that relate these conditions with the actions are representing procedural knowledge about the generalization process. This procedural knowledge together with the geometrical and structural knowledge is the foundation of the cartographic knowledge (Müller, 1991; Armstrong, 1991). It has to be formalized beforehand in order to be usable as rules defining the relation between conditions and actions and their order of processing. This definition of rules is also termed knowledge acquisition (Buttenfield and McMaster, 1991). The knowledge, however, is difficult to describe in a formal language (Harrie and Weibel, 2007) such that the rule-based approaches have to solve the knowledge acquisition bottleneck (Weibel et al., 1995). Therefore Weibel et al. (1995) propose a method for interactive process tracking to extract the cartographic knowledge from the interactions of human experts. These approaches are basically trying a reverse engineering of a manually executed generalization. The combination of constraint based evaluation and machine learning techniques for knowledge acquisition is shown by Mustière et al. (2000).

The *human-interaction* model is basically an interactive generalization model which is supported by more or less sophisticated tools for the execution of routine tasks such as cartometric analysis and basic generalization operators. The more complex cartometric and structural situation analysis and the control of the whole generalization process have to be carried out by experienced cartographers. Weibel (1991) proposed therefore the term "amplified intelligence" where the cognitive workload is shared between the computer and the human expert by combining tools such as simple rule-based systems with human interaction.

The notion of constraints[1] was introduced to generalization by Beard (1991). Constraints[2] designate a final product specification on a certain property of an object that should be respected by an appropriate generalization (Barrault et al., 2001). Thus constraints describe a goal which should be achieved by an optimal

[1] Constraints, as they are being used in mathematics or computer sciences, are conditions or restrictions on variable values (Tsang, 1993). In traditional Constraint Satisfaction Problems (CSPs) all constraints must be satisfied (or true) in the optimal solution. As an extension the Valued CSPs allow "hard" constraints that must be satisfied and "soft" (also weak) constraints that preferably should be satisfied (Bistarelli et al., 1999). Constraint base approaches in generalization make use of valued or weighted constraints having different priorities (Ruas, 1998).

[2] In this thesis the notion "constraint" refers to the context of generalization research (Beard, 1991; Ruas and Plazanet, 1996; Weibel and Dutton, 1998; Harrie, 2001) where also soft and hard constraints are allowed.

generalization. While measures are only characterizing objects or situations, without considering cartographic objectives, constraints are evaluating situations with respect to the formalized cartographic objectives. Thus the constraints check if the objects or situations are also in a cartographically satisfying state. The definition by Ruas and Plazanet (1996) focuses on the constraints related to objects while Weibel and Dutton (1998) see constraints as a more general design specification. In both cases constraints are evaluating object characteristics and are thus creating additional enriching information. In contrast to the use of simple rules, which are triggered by conditions and structural knowledge, constraints are not bound to a particular generalization operation. So, any number of different operations can be applied to resolve a constraint that has been violated.

Within the *constraint based* approaches, Harrie and Weibel (2007) differentiate *combinatorial* and *continuous optimization modeling* and *agent modeling*. In contrast Mustière (2005) differentiates two major categories: optimization methods and constraint based methods. But constraints can also be seen as part of an objective function used with optimization methods, making the two classifications compatible.

Combinatorial optimization techniques have been applied for example in the domain of label placement (Zoraster, 1986) and line simplification (Cromley and Campbell, 1992). Wilson et al. (2003) showed the use of genetic algorithms for the displacement of map features and Ware et al. (2003) used simulated annealing for the displacement of map features in combination with the option to delete, enlarge and shrink features. Combinatorial optimization is also used within this thesis for the control and chaining of a building generalization workflow consisting of completely independent generalization operators (see Research Paper 4 in §3.1.4).

Continuous optimization approaches are ranging from spring models (Bobrich, 1996), snakes (Burghardt and Meier, 1997) and elastic beams (Bader et al., 2005), finite elements (Hojholt, 2000) to least square adjustment (Sester, 2000; Harrie, 2001).

Constraints received particular importance in cartography through their use in conjunction with intelligent agents in the area of automated generalization (Ruas, 1998; Regnauld, 2001). The AGENT approach tries to minimize the constraint violations[3] for map features represented by autonomous software agents. These agents represent map objects and are trying to find an optimal state with low constraint violations (Barrault et al., 2001). Agent modeling is capable to handle the different types of generalization operations. Galanda (2003) demonstrated further the integration of optimization techniques within the AGENT model. Duchêne (2003) extended the agent model with capacities to perceive their spatial environment, as well as an ability to communicate with surrounding agents.

[3]The AGENT approach assigns a priority to each constraint. Thus this approach uses valued constraints (Bistarelli et al., 1999) that can be hard or soft.

2.1.4 Data Enrichment for Adaptive Generalization

The generalization process model by Brassel and Weibel (1988) underlines the importance of the detection of characteristic (termed structure recognition) as a basis for map generalization and formalizes the generalization process as a sequence of separate tasks. The model by McMaster and Shea (1992) mentions the importance of cartometric evaluation (roughly equivalent to structure recognition by Brassel and Weibel) in order to perform the appropriate generalization steps. Also Armstrong (1991), Ruas and Plazanet (1996), Mustière and Moulin (2002) and Regnauld (2005) stress the importance of structural knowledge for the understanding of the role of the features represented on a map and thus for automated generalization. This *structural knowledge* represents the spatial and semantical context of map features and their structural relationships. Thus, the context consists of the map features, their properties and of the relations between the features as well as between the properties. The representation and analysis of these relations, such as topological or metrical relations, are a fundamental part of GIScience (Worboys and Duckham, 2004). Raw spatial data can be enriched with semantical and cartometric measures as well as the knowledge about characteristic structures expressed by relations between map features. This auxiliary enriching information can be used for a variety of purposes within the overall generalization process (Weibel and Burhardt, 2003; Neun et al., 2004):

- Characterization: Map generalization seeks to maintain important map objects, patterns, and relationships, while suppressing unimportant ones. Hence, the spatial and semantic characteristics of map objects have to be detected in order to obtain priority orderings among map objects; meaningful groups of objects have to be formed (e.g., clusters of buildings, or objects aligned in a particular arrangement); and spatial and semantic relationships have to be detected (e.g., adjacency and proximity relations, hierarchical relations) before sensible and informed decisions can be made about generalization. It is important to note that such rich information is typically not available in spatial databases that are available for mapping, as these databases are usually designed to serve multiple purposes, and because comprehensive anticipatory object characterization would be far too costly.

- Conflict detection: Generalization is warranted if cartographic principles (i.e., cartographic constraints) are violated due to a scale change or a different symbolization. Conflicts are detected by comparing the results of the characterization step with thresholds imposed by cartographic principles (e.g., minimal size or distance thresholds for the target scale).

- Algorithm selection, sequencing and parameter selection: Depending on the character of the data to be generalized and the conflicts detected for the target scale, appropriate algorithms have to be selected and to be applied in the proper sequence in order to remedy the conflicts. Auxiliary data relating to the character of the data should help not only to select the most appropriate

generalization algorithm(s) from a range of potential candidates, but also to set appropriate values for the parameters that control these algorithms.

- Evaluation: Since the overall generalization process may involve many algorithms, it is critical that the success of individual and overall generalization operations is evaluated, particularly in systems with a capability of self-evaluation such as multi-agent systems (MAS) or evolutionary systems. Once again, the same methods can be used to generate the necessary auxiliary information as for continuous characterization, conflict evaluation, and algorithm selection and parameterization.

Figure 2.2 shows how the knowledge about a characteristic structure may influence the generalization of a group of small islands. Suppose the three dark grey islands in the left image are too small for being displayed properly. The upper two generalization solutions are ignoring the contextual situation. They either delete the three small islands or enlarge them individually until they have reached the minimum size. These solutions both meet the basic perceptual requirement (minimum size) but do not necessarily represent a good overall cartographic solution. The lower two pictures are showing more adequate solutions that are maintaining the typical structures or patterns by preserving the spatial arrangement as well as the size and shape relations. Such a solution holistic can only be obtained by consideration of relations between map objects.

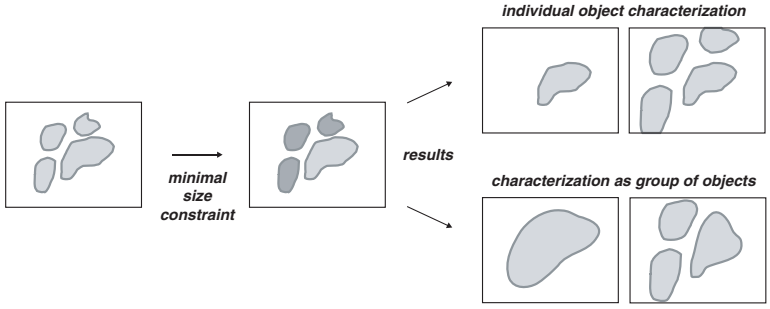


Figure 2.2: Generalization with and without additional contextual information (Steiniger and Weibel, 2007)

Structural knowledge represents the spatial and semantical context of map features and their structural relationships. This knowledge can be represented by a set of relations. Therefore Neun and Steiniger (2005) proposed the so-called *horizontal* and *vertical relations* for characterizing and modeling the knowledge used for the data enrichment. This differentiation is implicitly given by the map purpose and map scale. *Horizontal relations* of map objects exist within one specific scale, or level of detail (LOD), and represent common structural properties – e.g. neighborhood relations and patterns (Neun et al., 2004). On the left of figure 2.3 an example of a horizontal relation is presented by a curved polygon alignment. A detailed characterization of horizontal relations is given by Steiniger and Weibel (2007).

Vertical relations are links between single objects or groups of objects on different map scales or levels of detail (LoD), respectively (figure 2.3, right). The cardinality of such relations may vary between nullary, unary, and n-ary because of the fact that map objects on one level of detail have not always exactly corresponding objects on another level of detail. These vertical relations can be represented and stored in Multi-Resolution Data-bases (MRDB) for being exploited within paper map generalization as well as for web and mobile mapping applications. Adaptive Zooming is an example where previously extracted vertical relations (Cecconi and Galanda, 2002) can be used for achieving fast responding web mapping applications. Additionally Bobzien et al. (in press) introduced the so-called *update relations* for describing changes of map objects over time. This relation has three states (insert, remove, change) which can be used for example to insert a newly constructed building, remove a knocked down building or report a modification.

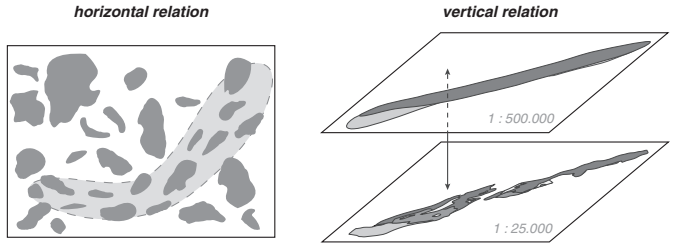


Figure 2.3: Horizontal and vertical relations between map objects and Levels of Detail

Motivated by the need of enriching information within the generalization process in Research Paper 2 (see §3.1.2) we introduced the term of support services as a generic way to model and provide the enriching knowledge (see also Research Paper 3 in §3.1.3). Thus, a main goal of such services is to make structural information explicit, representing common structural properties such as alignments, neighborhood or proximity relations, which may be usefully exploited by generalization operations. Support services can also be used for calculating measures and constraints. These two publications (Research Paper 2 and 3) are main contributions of this thesis and are thus discussed in section 3.1.

Characterization and Conflict Detection

It is interesting to note that many of the existing contextual generalization algorithms rely on auxiliary data structures. These algorithms are controlled depending on the spatial context in which they are applied (Regnauld, 2001). The spatial context is expressed by various measures and map object relations and is thus a subset of the structural knowledge. Usually is this contextual behavior implemented implicitly in the algorithms and thus the extraction and the use of the contextual knowledge are integrated closely. With data enrichment these formerly monolithic generalization algorithms are broken into functionally separate parts. In an initial enrichment phase the map objects are characterized with measures and attributes to

support the specific generalization algorithms with the required contextual knowledge. Once the auxiliary data structures have been generated, they may be used further by other algorithms and can thus increase the performance of generalization algorithms.

Geometric data structures that are useful to represent contextual topological and/or proximity relations include the Delaunay triangulation (DeLucia and Black, 1987; Jones et al., 1995), the Voronoi diagram (Gold, 1999; Hangouët, 1998), minimum spanning trees and other graph structures (Mackaness and Beard, 1993; Regnauld, 2003), skeletons (Bader and Weibel, 1997; Haunert and Sester, 2004), and hierarchical partitioning schemes (Ruas, 1995). Since they represent spatial relationships not originally represented in the "raw" spatial data these geometric data structures can also be seen as a form of data enrichment. Other data structures used in generalization research are for example multi-scale trees (Frank and Timpf, 1994), reactive data structures (van Oosterom and Schenkelaars, 1995) or quadtrees (Dutton et al., 1998). These structures lead also to the use of multiple representations or multiresolution databases, see for instance Egenhofer et al. (1994), Jones et al. (1995), Kilpeläinen (1997), Weibel and Dutton (1999), Spaccapietra et al. (2000), Kreiter (2002), Hampe et al. (2004), Dunkars (2004), Bobzien et al. (in press).

In the process of data enrichment for adaptive generalization, in a first step, the available data has to be analyzed for generating the enriching information. According to (Beard, 1988, 1991; Ruas and Lagrange, 1995) we can distinguish the characterization of geometric primitives and the modeling of spatial and semantic relations. In this thesis an extended taxonomy is proposed which differentiates supporting entities that contain geometries and attributes from supporting relations between map objects. The relations can be of spatial, structural or semantical nature and do have a hierarchical or non-hierarchical arrangement. An exhaustive review of relations used in map generalization is given in the Research Paper 3 in Part II of this thesis.

Supporting geometries can be used for many purposes during a generalization process. Examples include the computation of alignment lines, chaining together a group of map features such as buildings (Christophe and Ruas, 2002), the creation of inflection points or the identification of local extremes for line generalization (Plazanet et al., 1995). The techniques proposed by Plazanet (1996) and Mustière (1995) can also be used to isolate and characterize individual meaningful shapes, such as individual bends of a line.

Supporting attributes have been rather thoroughly investigated for many different purposes and object types. They can be for example measures that are calculated once and then saved as attributes. Examples of shape measures for line sinuosity include those based on distance relations between vertices of boundaries (McMaster, 1986; Dutton, 1999), based on the analysis of inflection points (Plazanet et al., 1995; Plazanet, 1996), or based on epsilon bands (Mustière, 1995). Other important attributes support the model-generalization operators such as classification, aggregation or association that require the consideration of spatial and semantic relations between objects (Ruas and Lagrange, 1995). An example thereof is the extraction of urban building structure classes as shown by Steiniger et al. (in press).

Evaluation of Generalization Results

An automated generalization process may involve many algorithms. In such a whole process from the initial to the final state there might be no human intervention to evaluate the cartographic quality. Thus also automated methods for cartographic quality assessment are needed in order to evaluate the generalization results. This evaluation capability is critical for the success of individual generalization operators and overall generalization processes, particularly in systems with a capability for self-evaluation such as multi-agent systems (Barrault et al., 2001) or continuously evaluating optimization systems (Ware and Jones, 1998; Ware et al., 2003; Wilson et al., 2003; Research Paper 4).

A comprehensive model for the assessment of the quality of cartographic generalization is presented by Bard (2004). He proposes the comparison of characterizations of initial and the final datasets and the aggregation of these various assessments resulting in a summarized level of generalization quality. Ehrliholzer (1995), Dutton (1999) or Brazile (2000) also address the issue of quality evaluation in map generalization, though on a mostly conceptual level.

Constraint based approaches are implicitly performing a quality analysis as long as all cartographic quality requirements are formalized as constraints. The individual constraint violations must be summarized with a *cost function* (Research Paper 4) or *evaluation function* (Bard, 2004) in order to be able to determine the overall quality. All cartographic quality assessment approaches rely on different measures and relate the map objects. This auxiliary information must be valuated in a qualitative or quantitative way and then summarized in order to assess the cartographic quality of a map dataset. Approaches for such a evaluation function are presented by Bard (2004).

Quality evaluation may occur at different stages in a generalization process. In some approaches only the initial state before and the final state after generalization are evaluated and compared (Bard, 2004). Other approaches such as the optimization techniques (Ware and Jones, 1998) require a complete evaluation of the current state after every step in a generalization process or even at every step of an algorithm. Purely constraint based approaches such as the one used in the AGENT project (Barrault et al., 2001) are evaluating the evolution of constraints continuously on an object and object group level.

2.2 Generalization Services

GIS and digital cartography has evolved in new directions due to the increasing use of web technologies. On the one hand there is a growing demand for on-demand or on-the-fly generated maps for the use as electronic maps both stationary and in a mobile context (Weibel and Burghardt, in press). On the other hand traditional map makers such as national mapping agencies (Regnauld, 2006) as well as the generalization research community (Badard and Braun, 2003; Edwardes et al., 2003) have expressed a demand for a common open research platform. Such a platform could serve as a universal toolbox allowing to share and use various algorithms and supporting methods.

For combining data from different data sources (Lehto, 2003) in systems for on-the-fly generalization or the research platform the use of structurally and semantically different data and systems must be possible in an interoperable way. This is also important for the coupling of different systems in a heterogeneous map making environment for example at a national mapping agency. The term interoperability is used to describe the capability of different programs to exchange data via a common set of methods or interfaces using the same protocols, and to read and write the same data formats. Interoperability can be achieved by using a services oriented architecture (SOA) with internet-based standards (Channabasavaiah et al., 2003). The Open Geospatial Consortium defines interoperability as "*software components operating reciprocally (working with each other) to overcome tedious batch conversion tasks, import/export obstacles, and distributed resource access barriers imposed by heterogeneous processing environments and heterogeneous data*" (OGC, 1996).

In the following section principles of SOA are introduced and an overview of recent developments within the domain of services for GIS and generalization is given. A review of computer architectures and standards with a special regard towards the use of services for an open generalization research platform is made in Research Paper 1 (see also §3.1.1) which is included in Part II. As co-author of this publication I contributed the development of the first prototype platform. Thus, this publication can also be seen as a motivation and starting point for the development of the services based WebGen framework which is presented within this thesis (Research Paper 2 and 3, see also Appendix A).

2.2.1 Interoperability and Distributed Environments

Up to the early 1990s software systems typically were closed, monolithic applications. During the 1990s, then, the way in which software systems were built and the way in which they delivered functionality to the user became more modularized, leading to component systems with reusable software components (Gamma et al., 1995) and to service oriented architectures (SOA). This was strongly influenced by the development of client-server architectures as for the World Wide Web.

Client-Server Architectures

The client-server architecture is a standard two-tiered architecture with two component levels: the client level and the server level. Components on the client level request services of components on the server level. The World Wide Web (WWW) employs this structure: The web browser on a client computer requests a web page from a web server residing on a remote server. The web server processes the requests and sends as answer the requested web page back to the client. The browser then receives the page and displays it. In more complex situations, an additional layer exists forming a three-tiered architecture. This middle-tier layer (also called middleware or business layer) analyzes the client request, makes itself a request to the application tier and sends then a response back to the client. Thus it acts as a sort of translator between the application tier (e.g. a database) and the client tier. The middle-tier can again consist of multiple layers and the architecture is then denoted as n-tiered (Ghezzi et al., 2003). Typical servers at the application tier are databases and legacy systems[4](Feiler, 2000).

Web Services as Services Oriented Architecture (SOA)

The traditional WWW client-server architecture is designed for a human-machine interaction where a user at the client computer requests a web page using the browser. Information is transferred using the HTTP protocol and presented in HTML. However, the information encoded in HTML is only understandable by humans while the computer only displays it without further interpretation. Thus this approach can not be used for the integration of different applications (Glass, 2002) in a distributed environment using automatic and self adapting machine-machine communication without human interaction.

Services oriented architectures (SOA) are an evolution of distributed computing with separate components using the request/reply mechanism for synchronous or asynchronous coupling of client and server. Application logics or individual methods are modularized and presented to the client applications as services having an implementation independent interface. These services interoperate according to a formal specification and can be accessed without knowledge of their underlying implementation (Channabasavaiah et al., 2003). Service interfaces are contracts between the service provider (server) and the service consumer (client) where the deliverables are detailed exactly but the implementation as well as the computing platform is hidden completely and can thus even be changed as long as it offers still the same interface. This separation of specification and implementation enhances the reusability of the modular services (Ghezzi et al., 2003).

Services extend the traditional approach of providing a software library with a well defined interface. Software libraries can often only be used in one particular programming language. Services, however allow the interoperable use with different

[4]legacy systems are existing “antiquated” computer systems which remain in use as they are difficult to replace or redesign even if they are hard to maintain and can not be developed further anymore.

languages on different platforms. Thus, services are more flexible, exchangeable and also comparable between different implementations. Valuable functionalities from different platforms can be combined allowing a user to flexibly request tailor-made services without having to care about the platform specific implementation and configuration. Finally, services can be accessed either over a network in a distributed environment or locally.

The initial and main goal in distributed environments is the ability to execute procedures and access data on a remote computer (see figure 2.4). This technology is also called remote procedure call (RPC). Implementations of RPC in common programming languages allow procedure calls without the programmer explicitly coding the details for this remote interaction and thus basically having no difference if executing local or remote programs. Within object-oriented programming languages RPC may be referred to as remote invocation or remote method invocation (RMI).

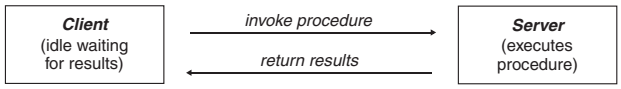


Figure 2.4: Standard Remote Procedure Call (RPC)

An SOA can fulfill the two needs of data exchange and also of inter-process communication in order to make a seamless communication between applications on different platforms possible. Therefore standard protocols and data encoding methods are needed that are widely supported on different computing platforms. With the advent of the World Wide Web the Hypertext Transfer Protocol (HTTP)[5] and the eXtensible Markup Language (XML)[6] were widely available and could be used to address the sharing of data and software functionalities over a network (Glass, 2002). With XML for the data encoding and HTTP for data transfer, difficulties caused by differences of computer hardware, operating system or programming languages can be prevented.

A first use of XML and HTTP for remote procedure calls was the XML-RPC technology (Winer, 1999). This approach allowed calling methods on a remote computer with standard data types like Integer, Double or String as well as arrays or

[5]The Hypertext Transfer Protocol (HTTP) (W3C, 1999) is the standard protocol of the World Wide Web. It is a generic, stateless application level protocol which is used to transfer hypertext files between clients and servers across a network such as the internet. Originally intended to transport hypertext files such as HTML it is also used for image files or any other type of text or binary data.

[6]The eXtensible Markup Language (XML) (W3C, 2006) is a cross-platform-, software- and hardware independent language for transmitting information. It can be applied to structure, store, and exchange information and is human- and machine-readable. In addition, XML provides with the Document Type Definition (DTD) or the XML Schema a self-descriptive mechanism. XML encodings are a special case of text-based encoding. XML documents exhibit a hierarchical structure with nested entities. This structure can be parsed for example with the Document Object Model (DOM) which represents the XML document as a tree in an object-oriented way. Examples for XML based data formats are GML or SVG which are used for the encoding or representation of geographical data as well as WSDL and SOAP which are used for web services.

binary data as parameters and also as results. XML-RPC evolved into the XML-based protocol SOAP (W3C, 2004). SOAP encapsulates the messages and describes how to process them (Englander, 2002). The messages usually contain the parameters of a service call or its results. The message content can be standard data types encoded in XML but also non-XML content such as binary data.

The so called web services are using common standards, having a common representation and offering interoperability (Curbera et al., 2001). Web services combine data and functionality making the needed information accessible using for example the SOAP protocol. Web services have a loosely coupled interaction model due to the heterogeneity of involved computer environments. The capabilities of a web service are described in a generic format like the Web Services Description Language WSDL (W3C, 2004) defining an abstract representation of a service (Riedemann and Timm, 2003). WSDL specifies the interface of a service in terms of the set of operations provided each with specific input, output, and error messages (Lake et al., 2004). WSDL provides a description of messages exchanged between a web service and its client in an XML Schema. Unfortunately WSDL is only capable to describe syntactic aspects of service capabilities (Riedemann and Timm, 2003), semantic aspects are not handled. With regard to distributed geo-services (Lemmens, 2006) proposes a formal semantics-based description of services. These semantic descriptions allow the computer-aided integration of distributed heterogeneous geo-information and geo-services.

In order to allow the individual use as well as the use of web services as components in, for example, a service chain a way of dynamically offering and retrieving services is needed. The so called "publish-find-bind" (see figure 2.5) paradigm is a model to support this need. It proposes a registry where services are registered (published) by their providers with their capabilities. Users seeking to use a service make a request to the registry in order to find the service and can then call it directly (bind). An implemented standard is the Universal Description, Discovery and Integration model UDDI (OASIS, 2002). However, unfortunately this retrieval standard lacks also semantic description capabilities that would allow truly dynamic and self-configuring service chaining.

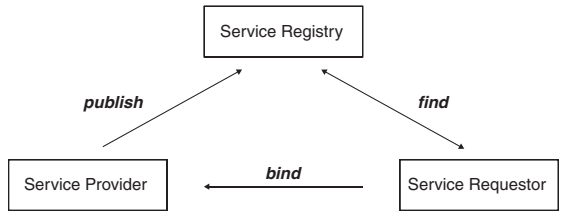


Figure 2.5: The “publish-find-bind” paradigm

2.2.2 Interoperability of Geographical Data and Algorithms

Within distributed environments the interoperability plays an important role. According to Vckovski (1998) there exist different aspects of interoperability such as independent applications on the same machine, independent applications exchanging data, and independent applications communicating seamlessly with each other even when they reside on different platforms.

Goodchild et al. (1999) state that interoperability in GIS deals with sharing information being distributed over different platforms, geographic locations, and database systems. Thus GIS interoperability development mostly focused on the distribution of data and metadata and neglected distributed operations (Conrad, 1997). The usage of distributed operations such as the sharing and accessing of distributed services, offering for example remotely invocable methods or programs, plays an increasingly important role (Goodchild, 2005). Nowadays the services allow using a much wider range of available operations in order to process the increasing data volume. A collection of universal geospatial operations is presented by Albrecht (1999). In order to speed up development and allow increased collaboration these geospatial operations are likely to be distributed as services allowing open access rather than being implemented on a closed single platform (Riedemann and Timm, 2003).

Open architectures based on distributed platforms have attracted significant interest in the generalization community (Lehto and Kilpeläinen, 2000, 2001; Harrie and Johansson, 2003; Badard and Braun, 2003; Research Paper 2). In the following two sections web service approaches for the delivery of (generalized) data, also termed middleware generalization services, as well as for the sharing and accessing of distributed operations, also termed ***generalization toolbox services*** (Research Paper 2 and 3) are introduced. An overview of technical generalization frameworks can additionally be found in Research Paper 1.

Data Delivery and Middleware Services

For achieving the data interoperability within GIS web services technologies are increasingly used. This focus of the development of (geo-) web services for data delivery is strongly driven by the desire to develop (national) spatial data infrastructures. In this domain standards are available which are widely used. The Open Geospatial Consortium (OGC, 2002) specifies these services such as the Web Mapping Service (WMS) or the Web Feature Services (WFS). These web services are usually extensions of databases containing the geographical map data. Every data server (see Figure 7) is connected to a network and provides its data in a standardized format. In the case of a WMS raster map data can be requested by defining the desired spatial extent. In the case of a WFS map features with geometries and attributes can be requested as OGC Simple Features (OGC, 1999) encoded in the Geography Markup Language GML (OGC, 2004; Lake et al., 2004).

With the application of web technologies such as WMS or WFS the cartography has evolved in a new direction, delivering and generating on-demand and on-the-

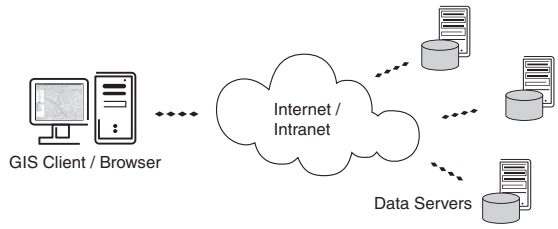


Figure 2.6: Data delivery services architecture

fly maps, containing more specific and tailor-made information. In contrast the traditional way of map making focused on the production of static (e.g. topographic) maps that were designed in advance for general purpose usage. The demand for dynamic web map generation has also lead to increased requirements on automated map generalization procedures (Cecconi, 2003). So-called middleware services can be used to deliver pre-generalized data originating for example from a WMS or WFS (Research Paper 2). An approach of this sort is described by Sarjakoski et al. (2005) using a modified WFS. Also, the OGC has expressed interest mentioning the *Feature Generalization Services* as part of their OGC Web Services (OWS) initiative (OGC, 2002). However, the specification process for such services has not experienced much progress so far.

A categorization of basic services for architectures portraying spatial information is described in the OGC *Portrayal Model* (OGC, 2003). It is also known as the *Cuthbert Model* (Cuthbert, 1998) and describes a pipeline of four sequential processing steps ranging from an initial filtering of features, over the application of styling rules and the rendering to the final displaying of the map. This model applies for the context of portrayal but can also be extended for generalization (Research Paper 1).

The Cuthbert model as well as other models for the transformation and delivery of data proposes a multi-tiered architecture. Thus, the services that take care of the intermediate processing or transformation are middleware services (see figure 2.7). Such a middleware service sits in the middle of the architecture, for example between client applications and data servers. As an example a client might request a generalized map from a generalization service. This service retrieves the data from a WFS, generalizes it and delivers it to the client. The generalization service acts as middleware between the client and the data source (a further example can be seen in figure 2.8).

Figure 2.7: Middleware architecture

The middleware scenarios are very well suited for the on-demand delivery of generalized data. This is because interfaces and protocols can be used that are standardized and well understood. Open source implementations of WMS and WFS do exist[7] and can be augmented with such middleware functionalities. Thus developers can focus on augmenting existing systems rather than starting from scratch.

Figure 2.8 shows the workflow of a service based generalization framework used in the GiMoDig project (Sarjakoski et al., 2005) for the delivery of real-time data integration and generalization in mobile cartography. In this architecture the mobile clients communicate with the portal layer which implements a WMS and delivers thus the final rendered map as raster graphic. The portal layer requests the generalized map data from the data processing layer which is an extended WFS. The generalization takes place in this layer. The data layer in this project consisted of several heterogeneous WFS servers that provided data from different national mapping agencies. The portal and the data processing layer are both middleware services translating between the data sources and the clients.

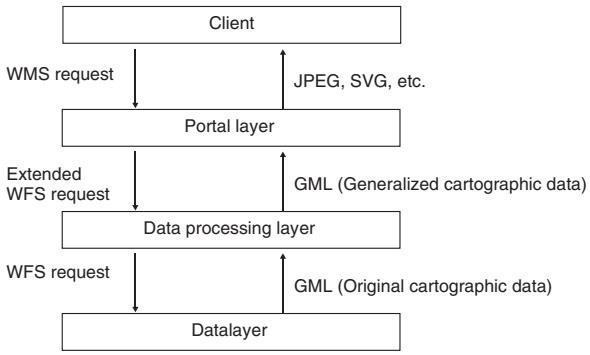


Figure 2.8: GiMoDig service architecture (from Harrie, 2005)

In the GiMoDig project XSLT[8] techniques were combined with the use of the Java programming language (Lehto and Kilpeläinen, 2000). XSLT is used for processing directly the XML based GML data. This is computationally fast but can only be used for simple tasks. Harrie and Johansson (2003) and (Tuvesson and Harrie, 2003) used the Java libraries JTS and JCS (Solutions, 2007) for implementing the more advanced and complex real-time generalization and integration algorithms. A framework based on XSLT and Java is also shown by Mannes (2004). This framework is used for the real-time generalization of point data animal observations. The generalization functionalities were developed as an extension to the WFS server implementation of the degree project (Fitzke et al., 2004). A commonality of most of the middleware approaches is that the data sources usually are one or multiple

[7]for example the deegree framework (www.deegree.org) or the GeoServer (www.geoserver.org)

[8]The Extensible Style Language Transformation is an interpreted programming language which is used to transform XML documents (e.g. GML) into other XML documents like SVG or also into other formats like HTML or PDF.

well defined databases and that the user does not need to be aware of where the data comes from. The clients usually communicate only with the middleware. This approach is well suited for fully automated processes where the complexity of the framework is hidden from the users. Thus these middleware frameworks are one-way services delivering data from the data source to the client.

Processing Services as Generalization Toolbox

Generalization algorithms as well as supporting data structures providing enriching structural knowledge are being developed by various research groups, typically on their platform of choice. The problem, then, is that although the generalization toolbox is seemingly steadily filling up no real progress can be made in research nor in production as long as these tools stay on different platforms (Edwardes et al., 2003). This is the motivation for a common open research platform (see also in the introduction §1.1.2 and in Research Paper 1 and 2) for generalization. The development of a services based architecture for the establishment of such a research platform is one of the main contributions of this thesis and thus presented in Research Paper 2. This section introduces the concept of *processing services* in contrast to the *data delivery* or *middleware services*. Additionally other (parallel) developments of generalization services are shown which were started during the duration of this thesis and which partially were influenced by the concepts presented in the Research Papers included in Part II.

Web Services are not only useful for accessing data over the web but also for accessing processing functionalities on a remote server in a platform independent way. In research but also in map production environments not only the interoperable chaining of services like in the middleware approach but also atomic access to generalization techniques is required. This can be supported by services that expose operations and their parameters in very detailed ways using a common interface (Research Paper 1 and 2). In this way interactive and automatic generalization operators as well as supporting methods and data structures can be used as a kind of universal distributed *generalization toolbox* allowing the coupling of algorithms and data structures on different platforms. Thus, the evaluation of algorithms etc. is facilitated. Generalization toolbox services can support research cooperation by sharing of techniques within the cartographic research community (Regnauld, 2006), for example new algorithms, enriching data structures or cartometric measures. Generalization toolbox services allow also the processing of computationally heavy procedures on fast servers instead of a local workstation.

In an open toolbox service model everybody may offer his/her own generalization services. Through the Internet and the use of platform independent technologies such services may reside on servers all over the world. The service model could also be used for the coupling of different platforms in a local intranet without exposing the services to the internet. The processing servers (cf. figure 2.9) are hosting the generalization services. The data to be generalized is specified or provided by the client. Thus, the user at the client has full control and can select services according to his needs.

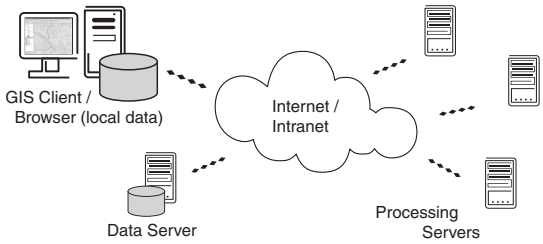


Figure 2.9: Processing services architecture

A ***generalization toolbox service*** offers its processing capabilities to the user including algorithm logics and computational power. The ability to provide specialized or novel algorithms in a platform independent way allows the evaluation and integration of generalization functionalities from different sources without forcing the users to adapt their systems to any specific needs. For such a full-fledged research platform, however, an interactive two-way data transfer is required from the client to the executing server and back. This scenario foresees that users can generalize their own data and parameters, by providing them to the service (cf. figure 2.10). The user may either include local data directly in the request or he/she could specify a link to a data server which is connected to the network (cf. figure 2.9) and where the generalization service is able to access the data. The generalized map is returned to the client as result (cf. figure 2.10).

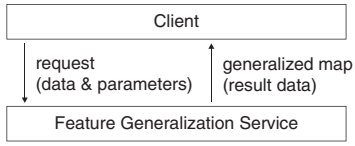


Figure 2.10: Processing services allow the upload of data and parameters

In order to initiate the process towards service-based computing in map generalization a possible starting point could be the development of small generalization services with narrowly defined functionality (ICA, 2004) without having to deal with the harmonization of data types and structures (Lehto and Sarjakoski, 2004; Sester et al., 2004; Illert and Afflerbach, 2004).

Several approaches for services based generalization tools and platforms have been developed recently. The first prototype of a generalization service platform was the **WebGen** framework. This prototype is a main contribution of this thesis and has been used extensively as a development platform for the DEGEN project (cf. §1.2.1). More details about the theory of framework can be found in Research Paper 2. The implementation of the WebGen framework is outlined in Appendix A. In the WebGen framework the generalization functions from different platforms can be offered as interoperable generalization services. Thus, they are available as large distributed toolbox. At the time of this writing 48 services from 6 authors

are available[9] in the WebGen framework. For the easy retrieval of services a central registry indicates which generalization services are available, where they can be accessed and what functionality they are offering. For every service a detailed service description describes the required input parameters and data schemata as well as the output. The WebGen platform consists of client and server components. Currently client plug-ins for the JUMP Unified Mapping Platform[10], a client as a browser based AJAX[11] web page and a prototype plug-in for the Clarity generalization software by 1Spatial[12] are available. A client with similar functionalities could also be developed for other desktop GIS having an API for adding functionalities. A Java based Server can offer Java algorithms as well as external programs as services. First experiments for providing services from within the Clarity generalization software were also successful (Regnauld, 2007).

Another approach for the services based generalization framework is shown by the project DRIVE (Burghardt et al., 005b; Bobzien et al., 2006). In this project the generalization functions of the cartographic GIS **axpand** of Axes Systems[13] were extended by a generalization service architecture, and a workflow management system. The web service architecture adopted the registry and service descriptions from the WebGen framework. The data, however, is not transferred over the network using XML and HTTP but accessed directly from a multi-representation database. Therefore the requests to the services contain only a link to the data which has to be processed and which is then retrieved by the services themselves using e.g. SQL for querying the database. This approach works well in a closed local map production environment but the services can not be offered for collaboration and exchange over the internet as they require the local database. Conversely the axpand system is capable of using services that are offered as WebGen services. The workflow management system allows the graphical design of workflows consisting of all possible generalization services and workflows offered by the registry server. The workflow engine is part of the generalization server. So, the generalization server can execute workflows as well as single generalization services.

The **OXYGENE**[14] platform (Badard and Braun, 2003) developed at the COGIT laboratory of IGN[15] is intended to provide an interoperable framework for the development of geographic applications and the coupling of applications on different platforms. The platform is based on an object-oriented schema implemented in Java for the representation of geographic information and implements OGC standards.

[9] see updated listing at http://webgen.geo.uzh.ch

[10] JUMP Unified Mapping Platform, http://www.jump-project.org

[11] AJAX stands for Asynchronous JavaScript and XML. It allows a Web page to communicate with a server exchanging small amounts of data without reloading an entire Web page. Thus user interfaces become more responsive. A well known example are the Google Maps, http://maps.google.com

[12] 1SPATIAL Inc., Radius Clarity Software, http://www.1spatial.com

[13] Axes Systems AG, Automated GIS Cartography Solutions, http://www.axes-systems.com

[14] OXYGENE later evolved into the open source framework GeOxygene, http://oxygene-project.sourceforge.net/

[15] Institut Géographique National, the french national mapping agency, http://recherche.ign.fr/labos/cogit

Interoperability between the OXYGENE platform and applications written in other programming languages is achieved from within the object-oriented schema using the Java Native Interface[16]. OXYGENE is intended for the deployment of geographic processes in general. The usage of SOAP and WSDL allows the interoperable use and the syntactical description of basic service interfaces. This use of web service standard protocols allows the use of existing software libraries, an extension however with more geo-specific features and semantical information is difficult. The workflow with an example service method is shown in figure 2.11.

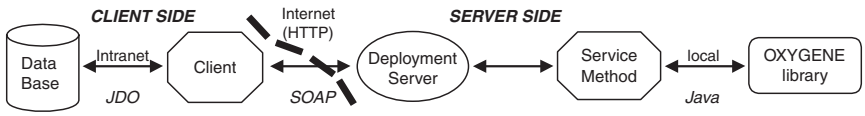


Figure 2.11: OXYGENE Geographic WebService – typical workflow (from Badard and Braun, 2003)

Besides the above mentioned Feature Generalization Services (OGC, 2002), which are targeted more towards a middleware architecture, the OGC proposed an implementation specification for a generic geoprocessing service (OGC, 2005; Schut and Keens, 2005), the so called Web Processing Service (WPS). The **WPS** specification is a first step towards web services for the execution of spatial algorithms, including those for generalization. This specification, however, is still a proposal and is awaiting its revision and completion since the call for public comments in January 2006. A response to this call for comments can be found in Appendix B. In the proposed state the WPS specification is very unspecific and open as there are no strict definitions of the data format nor the data schema or semantics. As a consequence, different WPS for generalization might not be compatible. Further activities towards the specification of generalization web services are therefore required. The WPS specification harmonizes the standard web services concept consisting of SOAP, WSDL and UDDI with the OGC web services like WMS or WFS. Similarly the specification foresees three basic methods for each WPS. The *GetCapabilities* method lists available operators on a server, the *DescribeProcess* method returns the syntactical description of a specific service and the *Execute* method performs the specific operator on the server. As for the other service approaches XML and HTTP are used for the encoding and transfer of data and parameters. Clients for accessing a WPS might also be implemented for different GIS applications. Foerster and Stoter (2006) present a first WPS prototype implementation for the remote execution of generalization operators using the JUMP software[17] as client. Lemmens et al. (2006) propose the integration of semantic and syntactic descriptions in order to allow to chain geographic services. Their proposal is not specifically made for the use of WPS in generalization but applies nevertheless.

[16]SUN Microsystems: The Java Native Interface, http://java.sun.com/docs/books/jni/
[17]JUMP Unified Mapping Platform, http://www.jump-project.org

Another web service approach is aimed at more interactive generalization workflows. Harrower and Bloch (2006) present an interactive web service for browser based line simplification called **MapShaper**[18]. The client side of this service approach is completely browser-based. The user sends the data to be generalized to the server via an upload functionality in the browser. Then the line simplification can be done interactively, while the data is processed on the server. Finally the generalized data is downloaded and exported to the client computer again. This workflow is shown in figure 2.12. Other than the previously shown service approach MapShaper is aimed at providing purely interactive user controlled generalization functions instead of aiming at interoperability and coupling of different generalization systems for automated generalization. Thus this approach has some limitations for its use in a real generalization scenario. The client for MapShaper uses Flash[19] and runs in a browser. Therefore the service can only be used standalone controlled by a user to generalize a (small) dataset but it can not be used in an automated workflow using larger data sets and incorporating several services. The MapShaper software is currently designed for providing line simplification functionalities. Any addition of other generalization functionalities like with polygons would require new versions of the client software in Flash as well as on the server making this approach not very generic.

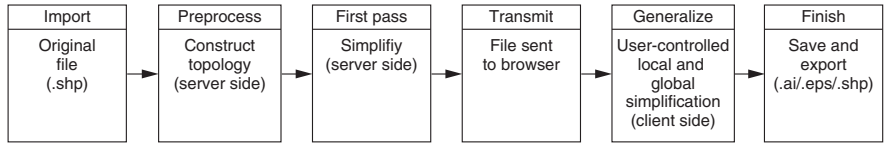


Figure 2.12: MapShaper.org Generalization Web Service – typical workflow (from Harrower and Bloch, 2006)

Finally, processing services in general are aimed at the provision of processing capabilities with interactive or automated tools on the other hand the middleware services are more intended to do a on-the-fly processing for the delivery of data. This overview, presented in the preceding sections, gives an impression of the heterogeneity of generalization frameworks inside the generalization research community. It shows and justifies the growing interest into services driven generalization research. As a last remark of this state of the art it must be mentioned that category of the processing services after the middleware services are not only of interest for map generalization. Thus there is also a growing interest of the GIS industry into service approaches like the server based processing. An example is the recently introduced ArcGIS Server[20] allowing to serve toolboxes containing one or more geoprocessing tools and their use e.g. from within a browser based environment.

[18] MapShaper shapefile editor, http://www.mapshaper.org

[19] Flash by Adobe allows the use of vector graphics to create animations and interactive applications that can be viewed by any browser having the Flash plug-in, http://www.adobe.com/products/flashplayer/

[20] ArcGIS Server, http://www.esri.com/arcgisserver

2.3 Summary: Challenges for Research

This chapter provided an overview about latest developments in map generalization and an introduction in the use of services-based technologies for map generalization. The objectives and research questions stated in section 1.2.1 were, of course, highly influenced by the topics discussed in this state-of-the-art. Thus, below are summarized the main challenges that triggered the research for this thesis:

- Map generalization should emphasize the essential structures and suppress the unimportant details (Weibel, 1997a). The automation of map generalization, however, is a complex process. The "holy grail" would be an autonomous self-evaluation system (Mackaness, 2006). Typically various methods (e.g. the ones proposed by McMaster and Shea, 1992) are used in a generalization process. The term adaptive generalization describes the goal to select the right methods (generalization operators) and the right parameters for the given map situation. The ultimate solution would be a generic "generalize (map)" function which takes an input map and delivers a generalized map with any desired scale as output. This challenge is an overall motivation and main driver for research on the automation of map generalization.

- Many different approaches of variable sophistication and applicability exist for the automation of generalization processes (Harrie and Weibel, 2007). Most of these approaches rely on additional advanced structural knowledge for controlling the generalization process. This auxiliary enriching information can be used for the characterization of map objects, for conflict detection, for algorithm and parameter selection and finally for the evaluation of generalization results (Weibel and Burhardt, 2003; Neun et al., 2004). The domain of data enrichment is not clearly defined: enriching data is often only implicitly implemented by the developers in their algorithms rather than being represented and extracted explicitly from the 'raw' input data in a re-usable form. Usually the extraction and use of the structural knowledge are closely integrated with particular generalization algorithms and they use proprietary data models or formats. Therefore generic methods for the use and re-use of the enriching information would be helpful.

- The development and exchange of generalization algorithms and data enrichment methods between researchers is difficult due to the heterogeneous research platforms used in the various research groups. Thus, a need for a common research platform has been expressed by the generalization research community (ICA, 2004). The development of a common open research platform that would allow testing and sharing of generalization algorithms (Badard and Braun, 2003; Edwardes et al., 2003) for a better progress in generalization research.

- In the GIS community the terms of interoperability and web services are usually only used about the sharing of distributed information as stated by Goodchild et al. (1999). But services can not only be used for the provision of

data, they can also be used for providing processing capabilities on remote servers in a distributed environment. Such processing services can provide the interoperability between applications on different systems. In this way interactive and automatic generalization operators as well as supporting methods and data structures could be used as a kind of universal distributed generalization toolbox allowing the coupling of algorithms and data structures on different platforms and allowing also the offering of generalization functionalities externally to the growing number of geospatial services on the internet. Such a toolbox could also serve as an open research platform for generalization research.

As an answer to the above challenges within this thesis a services-based framework, consisting of support, operator and process services, is proposed. Through the support services, this framework is capable of providing enriching information to generalization operator and process services in an interoperable way. Thus this thesis intends to prove the following two hypotheses:

1. Data enrichment helps generalization and makes thus adaptive generalization possible.
2. It is possible to build a fully fledged, interoperable common generalization system that is based on a distributed services oriented architecture and that allows the provision and exploitation of data enrichment.

Chapter 3

Results and Discussion

The five Research Papers included in Part II represent the main substance of this thesis. As a comprehensive access guide for the reader, this chapter sets the frame of this research and summarizes and discusses each paper in a broader context and offers a structured account of the research accomplished. These summaries, however, do not provide a substitute for the reading of the full papers.

The work presented in the Research Papers is original research carried out by myself. For papers where I am not first author, my contribution will be explained in the summaries given in the following sections (§3.1.1, §3.1.2, §3.1.5).

3.1 Synthesis of the Research Papers

The Research Papers are ordered in a logical sequence showing the complete framework of web service approaches for the provision and exploitation of enriching information within map generalization. The three main **objectives** derived from the rationale of this thesis were:

1. Storage and provision of enriched data using ***generalization services***
2. Identification and formalization of structural relationships for ***data enrichment***
3. Exploitation of enriched data for ***adaptive generalization***

In order to accomplish these objectives a couple of **research questions** were to be answered:

(a) What are the requirements for an open research and development platform?

(b) Can a services based architecture be used for the sharing of algorithms and enriching information in an interoperable way?

(c) What types of structural knowledge can be provided by services for the support of generalization methods?

(d) How is the knowledge about map object relations being generated and used?

(e) Can this knowledge about relations serve as enriching information?

(f) How can an adaptive generalization process be modeled using a framework of generalization services?

(g) How can the enriched data be exploited in order to control an adaptive generalization process?

This allows making associations between the objectives, research questions and the presented Research Papers (see also table 3.1):

Research Paper 1 (§3.1.1) advocates the development of an open generalization research platform. The requirements of such a platform are discussed and two open architectures are described and illustrated with first experiments (research question a). This paper does not address a particular objective given above, yet provides a motivation for the thesis.

Research Paper 2 (§3.1.2) describes the development of a classification and framework of generalization web services (research question a) and the technical implementation of the prototype generalization services framework called WebGen. This platform is intended to serve as a proof of concept for a common research platform for generalization. With this platform the provision of enriching information (objective 1, research question b) was accomplished.

Research Paper 3 (§3.1.3) focuses on the category of the support services providing the enriching information (research questions c and d) to generalization operators and processes (research question b). Thus it treats the classification and modeling of structural knowledge (objective 2) especially expressed through relations (research question e).

Research Paper 4 (§3.1.4) shows the development and experiments with process services for the automatic chaining of generalization operators (research question f). Thus it demonstrates the control of the generalization process with the help of enriching information provided by support services. These processes provide a successful exploitation of structural knowledge (objective 3, research question g).

Research Paper 5 (§3.1.5) extends the process service approach from Research Paper 4 and uses a knowledge base with previously successfully executed operator sequences for speeding up the operator chaining in the process control. This knowledge base relates the sequences with the cartographic constraints, they solved. Thus this information represents also enriching information for the control of the generalization process (objective 3, research question g).

Research Paper	*Objectives*	*Research Questions*
1		a
2	1	a, b
3	2	b, c, d, e
4	3	f, g
5	3	g

Table 3.1: Overview of the relations between Research Papers, Objectives and Research Questions

3.1.1 Research Paper 1: Experiments to Build an Open Generalisation System

Edwardes, A., D. Burghardt and M. Neun (2007). Experiments to build an open generalisation system. In W. Mackaness, A. Ruas, & T. Sarjakoski (Eds.), *Generalisation of geographic Information: Cartographic Modelling and Applications* chapter 8, (pp. 161-175). Amsterdam: Elsevier Science.

The demand for generalization processing capabilities is steadily growing and the separately developed generalization methods are getting increasingly complex. Therefore researchers started to investigate how open architectures might better support the various new geospatial services and technologies with generalization capabilities. An open research platform would allow closer integration in terms of collaborative research, data abstractions, interoperability of functional components and the augmentation of geo-spatial applications with generalization capabilities. This desire has been evidenced through discussions at the various meetings of the ICA Commission on Generalization and Multiple Representation (Beijing 2001, Ottawa 2002, Paris 2003, and Leicester 2004) as described in Edwardes et al. (2003).

One of the two case studies contained in this Research Paper reports on the WebGen framework which is a key contribution of this thesis. This book chapter is included in this thesis in order to introduce the need for generalization web services and thus to motivate the development of the WebGen framework in the context of the requirements of the generalization research community.

Contributions

Influenced by several efforts for an open generalization architecture (see Lehto and Kilpeläinen, 2000, 2001; Harrie and Johansson, 2003; Badard and Braun, 2003; Re-

search Paper 2) the book chapter discusses the issue of openness in relation to the requirements of the generalization community. Thus, as first main contribution, this Research Paper reflects on requirements for the sharing of generalization techniques amongst researchers as well as for presenting generalization functionality externally to geographic services. In the broader context, open standards, open architectures and open source approaches for geographic information are discussed. Therefore existing standards for the encoding of geographic data as well as for the achievement of interoperability between different platforms are reviewed in terms of their applicability for generalization research (research question a).

After this extensive review the Research Paper contributes case studies of two alternative architectural models for an interoperable access to generalization techniques. This categorization reflects the advanced characteristics for generalization services presented in Research Paper 2 and differentiates middleware services for on-the-fly generalization and an interactive generalization platform which may be used as open generalization system.

Based on the requirements of the generalization community the two categories are discussed as case studies in order to give an overview of possible architectures. The first approach consists of a middleware component that presents generalization functionality through a coarse, web mapping interface. It reuses existing protocols for requesting maps or map specifications and it is based on open source technologies for its core infrastructure. As second architecture the prototype of the WebGen framework is shown which adopts the model of a web service framework that provides access to generalization operations for the remote processing in a distributed and platform independent way. This framework is described in more detail in Research Paper 2 (§3.1.2). Technical details can be found in Appendix A.

Discussion

The research demonstrated in the two case studies (prototypes) of this publication is addressing the issue of improved access to generalization techniques. In each case the use of standards and frameworks from the existing body of knowledge on architectural design was found to be highly desirable. This was partly for practical reasons, as it helped to keep research efforts focused on the supply of auxiliary tools for generalization, by providing technology neutral, standardized design concepts and protocols that made it easier to express where and how functionality could be deployed.

However, the use of common standards for data transfer and interface descriptions lacks concepts and methods required to undertake and share very complex generalization research and to describe the services capabilities as well as the needed input and output sufficiently. An example is the modeling of highly contextual geometric relationships or the construction of sophisticated control structures for managing sequencing and orchestration of operations. Instead of handling cartographic constraints implicitly by parameters these would ideally be specified independently in a more formal way as separate constraints (c.f. Edwardes and Mackaness, 1999; Hardy et al., 2003; Sarjakoski et al., 2005).

Also the wider issue of semantics in data and parameters needs to be addressed. Generalization research often needs ontological descriptions (Gruber, 1993) about geographic phenomena. For example, the algorithm of Rainsford and Mackaness (2002) for generalizing rural buildings needs to make explicit definitions about what constitutes a rural building for their algorithm. Without such descriptions, assumptions must be made which lead to difficulties in reusing techniques if the two parties (designer and re-user) do not share a common understanding over the definition of feature type. These interoperability issues in terms of exchanging data with semantic translations (Kottmann, 1999; Kuhn, 2005) are not only important for generalization research but for all GIS areas, including national spatial data infrastructures (Crompvoets, 2006).

In generalization research a consensus on the formalization of the interoperability requirements is needed which can on the one hand be resolved through standardization (OGC, 1999) but also by practical experience in sharing research. In this regard the presented case studies, including the WebGen framework have strong contributions to make. Its relative simplicity, language neutrality, and ease of publishing and accessing algorithms means the barriers to researchers interoperating their algorithms are very low. Thus, research results can be shared fairly rapidly. In addition, because the owners of each algorithm remain in complete control of its source code and the responsibility for its maintenance, intellectual property obstructions and administration issues are reduced and cooperation between commercial and non-commercial parties more feasible.

3.1.2 Research Paper 2: Generalization Services on the Web – Classification and an Initial Prototype Implementation

Burghardt, D., M. Neun and R. Weibel (2005). Generalization Services on the Web – Classification and an Initial Prototype Implementation. *Cartography and Geographic Information Science*, Volume 32, Number 4, October 2005, pp. 257-268.

This paper aims to show advantages of applying the concept of a services oriented architecture to generalization, proposes classifications of generalization services at different levels of granularity and demonstrates a particular architecture. A first version was presented by Dirk Burghardt at the Auto-Carto 2005 conference (Burghardt et al., 2005a) where it was selected to appear in a special issue of the journal *Cartography and Geographic Information Science*. Therefore the first author is Dirk Burghardt although the contributed work to the paper was equal.

Contributions

Generalization services may be used in different application scenarios, for instance as a middleware component to allow on-the-fly adaptive zooming, or as an interactive generalization service which provides a toolbox. This generalization toolbox service approach allows the development of a common research platform, where researchers would have access to a common generalization framework and where algorithms and

supporting data enrichment methods could be shared (research question a). It could also be used for the coupling of different platforms in the production of topographic maps by national mapping agencies.

Three basic categories of generalization services were identified being *support services*, *operator services* and *processing services* (cf. figure 3.1). *Support services* can be seen as a provider of additional (enriching) cartographic information (research objective 2) in support of the automated generalization process. *Operator services* deliver the functionality of standalone generalization operators such as the ones defined by McMaster and Shea (1992). *Process services* use services from the lower categories for the control and guidance of a generalization process using operator and support services.

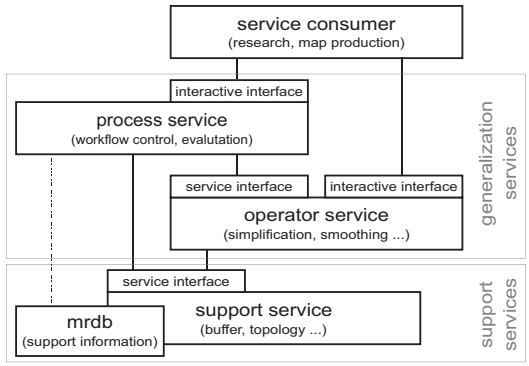


Figure 3.1: Categories of generalization services

As required by the research objective 1 the support services provide a way for the storage and provision of enriching information (research question b) for the support of the generalization services (operator and process services, figure 3.1). These support services might also be of use for other operations than generalization (Research Paper 3). The exploitation of the support services (objective 3) by process and operator services is addressed in the Research Papers 4 and 5.

This subdivision into the three service categories allows the functional decomposition of the generalization operators and processes. The services are modular and can be recombined and thus reused for different generalization tasks. For instance, a specific support service can provide enriching input to different operator as well as process services (see Research Paper 3 for more details; §3.1.3). Furthermore, different processing strategies, implemented as services, can use the same operator services and can rely on support services for the decision making process during a generalization workflow (see Research Papers 4 and 5 for more details; §3.1.4 and §3.1.5).

In an attempt to stimulate the discussion about generalization services in the generalization research community this paper explained the minimal requirements for the client and server side implementation of a *generalization toolbox service*. In this context the WebGen framework is presented in this publication as a prototype

of a research platform implemented as a generalization toolbox service based on standard web service technologies.

In the WebGen framework the generalization functions can be offered as generalization services on a generalization server. Several generalization servers could exist, due to the platform independent web service technologies, even on different platforms, offering a variety of generalization services. Thus the services can then be accessed independent of language bindings or of the location where the code is being executed. In order to facilitate the finding of available services a central server, the so-called registry server, indicates which generalization services are available, where they can be accessed and what functionality they are offering. For every service a detailed service description is provided, describing the required input parameters and data schemata as well as the format of the delivered output. Thus it is shown how developers and researchers can make their generalization functionality available by means of an interface description and how possible users of these services can find them through a registry database. Standards based encodings, such as GML for the geometry data, are used. This allows an appropriate definition of parameters and data types required by the services.

Discussion

The functional decomposition into the three different service categories has many advantages as it offers modularity, reusability, portability as well as interoperability. To take full advantage of the modularity further research is required on the segmentation of generalization tasks into the different service categories. For some cases this might also result in an overhead as the separate services are only loosely coupled and have to communicate through the standardized interfaces. Sometimes algorithms and data structures are very closely related such that their functional decomposition is not possible. Thus these data structures can not be offered as support services. The whole algorithm, including the data structure, could however still be offered as a service at a coarser level of granularity. A still largely unsolved problem is the automated orchestration of generalization operators. Thus, further research has to investigate ways of interacting and chaining of generalization services. These approaches can then be realized as processing services (see also Research Papers 4 and 5; §3.1.4 and §3.1.5).

The advantages of the generalization toolbox services are their availability and interoperability. This means that in the research community many people can use the services as long as their particular platform has a plug-in to call the service. Services of the different categories (support, operator services) can be combined simply by a calling client or as processing services. Professional users, such as map producers, may use the services for building prototypes that reuse and combine existing services. They would then have to develop only really new missing functionalities. Thus, algorithms can be evaluated for their aptitude in map production without having to implement them first in the production systems.

The support services allow portability due to the provision of tools and libraries that are otherwise not available for a specific platform. Due to this availability

of a rich set of support services the researchers are able to benefit from a quicker prototyping of new ideas. More time can be spent on developing new innovative functionalities in algorithms instead of having to redevelop basic functionalities that are needed as support.

Technical limitations of generalization services include first of all the transfer times. An overhead occurs as all data and parameters have to be encoded and transferred over a network from the client to the server in a web services environment. With very large amounts of data this might be too costly but as the network performances are still evolving the performance barrier will continuously be pushed forward. Another possible solution for the network performance limitation could be the shipping of code or of platform independently executable bytecode instead of the shipping of large data volumes. But this code shipping would then pose the question whether the code is really executable on another platform and how the copyright and licensing can be assured. Furthermore, current algorithms often rely on quite heavy software libraries which must be transferred with the code such that the code to ship might be very large as well. In a production context also the service availability is a serious constraint as an external server which is not working could interrupt an entire workflow. The WebGen framework is designed as a platform for research and development. Here these issues (transfer time and availability) are not that critical as the solutions usually serve as demonstration prototypes for methods which, if they have proven their aptitude, can afterwards be implemented in the core of a production system. Thus the client calls a service and waits until its completion. For more time consuming operations it would be possible to create a system which informs the user when the service has completed and allows the client to do continue other tasks or even start other services simultaneously.

In the context of managing an entire generalization workflow or accessing a completely server-based generalization system, solutions for extended session management and data persistence on a server must be considered. Such a stateful behavior could also be a way of flexibly accessing more complex systems as data and parameters remain on the server for subsequent calls. Furthermore data transfer bottlenecks with larger data volumes can be prevented. This issue will be addressed in Research Paper 3 which is summarized in the next section 3.1.3.

The generalization toolbox service model foresees the availability of client plug-ins for different GIS or map production platforms. Such a plug-in acts as an interface between the local data format and routines and the standard data format used by the service. Such a plug-in has to be implemented for every platform but once it is available a very large library of services exists for the platform without any new implementation work. At the time of this writing 48 services from 6 authors are available in the WebGen framework. Also the use of more traditional software libraries would often require the implementation of some sort of interface in order to make it compatible with one's own code. Nevertheless, this need of a plug-in might remain a constraint in some cases.

Also the required adoption of an existing algorithm as a service might pose problems if very special input or interface capabilities are required that can not be provided with the standardized services interfaces. Up to now such problems did

not occur and there probably will not be many cases as usually all algorithms follow some conventions of software engineering imposed by their programming language. If a new algorithm is implemented as a service this constraint does not apply.

The interface capabilities must be describable by the service descriptions. The web service standard WSDL allows only syntactical descriptions of the service in- and outputs. Therefore the service descriptions in the WebGen framework use a new format which also allows some more specific metadata, including the feature schema of the input data, in order to describe the generalization service. However, this is still not sufficient and therefore more research on the semantics in data and parameters is required. In order to allow the automated orchestration of different services the service descriptions should also provide information about the granularity of the service and its dependencies on other services. Furthermore the service's capabilities should be described in a generic way (Lemmens, 2006). This could be accomplished for example by an ontology containing service capabilities and dependencies as proposed by Regnauld (2007).

Open issues concerning the transfer and use of advanced data structures are how protocols can better express more contextual inter-object and inter-class spatial relationships as well as the semantic relationships between features, including priority ordering and attribute similarity. The structures containing information about these relations are enriching data and must be made available to the services. Such structures require often special data formats and are heavy to compute, thus they should also be made persistent. The management and transfer of such data structures is addressed in Research Paper 3 which is presented in the next section 3.1.3.

3.1.3 Research Paper 3: Web Service Approaches for Providing Enriched Data Structures to Generalisation Operators

Neun M., D. Burghardt and R. Weibel (in press). Web Service Approaches for Providing Enriched Data Structures to Generalisation Operators. Accepted for *International Journal of Geographical Information Science*.

This paper focuses on the category of support services which was initially introduced in Research Paper 2 (§3.1.2). The paper identifies and formalizes structural relationships which can be used for data enrichment (objective 2). In a service-based generalization system, the purpose of support services is to assist the generalization process by providing auxiliary measures, procedures and data structures that are difficult or expensive to calculate and that allow to represent structural cartographic knowledge (objective 1). Thus, the support services are providing data enrichment for later use by operator and process services. The aim is to make support services available to developers of generalization algorithms in such a way that they can use these in conjunction with generalization operator services without having to know in detail how the support data is generated.

Contributions

Structural knowledge about the spatial and semantical context and the modeling of structural and spatial relationships is critical for the understanding of the role of cartographic features and thus for automated generalization. Support services should extract and model this knowledge from the raw data and make it available through a standardized interface. Generalization support services can then form the foundation of the more advanced generalization operator and generalization process services.

After a brief introduction to the interoperable web service framework, this paper delivers as first contribution a comprehensive taxonomy of generalization support services (cf. figure 3.2) in relation to different generalization operators. The taxonomy distinguishes between enriching *entities* and enriching *relations*. On the one hand the structural knowledge can be expressed by enriching map features (entities) with additional geometries or attributes. On the other hand there exist various hierarchical and non-hierarchical relationships between map features (relations), many of which can be represented by graph data structures.

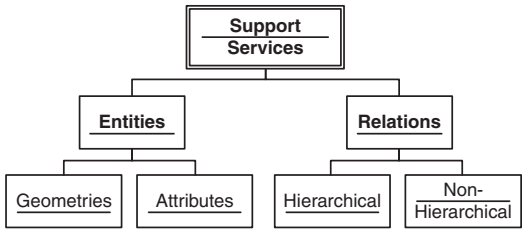


Figure 3.2: Support service types: Entities (simple attributes) and Relations (complex data structures)

The enriching *entities* can be expressed by simple attributes. Thus no new data formats for the storage and transfer are needed. A data structure like the Simple Features (OGC, 1999) is completely sufficient. The *relations*, however, do usually need more complex data structures.

Many methods for the enrichment of spatial data with structural knowledge are well known or even already implemented but they are usually implicitly contained in the specific algorithms or use proprietary data models and formats (research question c and d). Support services are making these algorithms and data structures available in an explicit form for the use in the generalization process in a web service environment. Thus, as a second contribution methods were proposed to represent, store and exchange the spatial relations generated by support services, considering the special requirements of distributed architectures (research question b). Since many relations can be expressed in a graph-like structure the proposed data structures and formats are mainly graph based.

Finally, the utilization of support services is illustrated on four implementation examples of generalization support services in the WebGen generalization service framework (research question e). It was demonstrated how generalization support

services can interact with generalization operator services, delivering complex data structures to complex algorithms. These examples demonstrate possible usage scenarios and benefits of using support services. They have been selected so as to provide a representative sample of the functionalities and data structures employed by the proposed taxonomy of generalization support services. As an example for non-hierarchical relations a stateful service for triangulations and proximity relations is shown which is then subsequently used for building typification and displacement. As an example for structural relations, expressed by non-hierarchical relations, a service for the detection of building alignments is shown. As an example for spatial and semantical context represented hierarchically or as a decision graph a service for the reclassification of features and their schemata is described. The final example presents a service for supervised classification of buildings into urban structure types.

Discussion

The aim of this paper was to illustrate the possible usage of generalization support services as a functional component which provides enriching input to generalization operators and for the control of the generalization process. Obviously, there are still open problems remaining, as the delivery of generalization capabilities is revisited, facilitated by a different architecture (service-based rather than standalone) and, at least partially, new constraints do occur including the processing efficiency with many data transfers between single services.

Standardized functionalities to compute and use spatial and structural relationships are sparse in GIS packages, particularly in commercial ones. Furthermore, geographic databases or data transfer formats do not commonly include the modeling and storage of the advanced relationships discussed in this research paper. We propose ways to use this knowledge in a web services environment. The WebGen architecture is not intended to be a web service for managing, browsing and exchanging data over the Internet but a processing service for providing algorithm logics and processing power to be executed remotely. It is, in contrast to other commercial or non-commercial GIS geo-processing servers, not limited to a (proprietary) GIS platform. The usage of an open protocol and standards such as XML and HTTP enable the interoperable coupling of service providers and consumers.

With a services based architecture the formerly monolithic generalization algorithms are functionally and technically subdivided. The support services are then performing the initial enrichment phase where objects are characterized with measures and attributes. This step supports then the specific algorithms with the needed contextual knowledge. Once the auxiliary data structures have been generated they can further be used by other algorithms and can thus increase the performance of generalization algorithms. Support services can also serve to evaluate results of a generalization process by calculating and analyzing measures and cartographic constraints and can therefore be used for the control of the generalization process (cf. Research Paper 4; §3.1.4).

The use of distributed web services for providing supporting data structures might not always be appropriate: when working with extremely large data volumes or high frequencies of service requests the process performance might be poor due to network latency, bandwidth and the time needed for creating and parsing the XML messages. In a *stateless* service implementation the entire data has to be transferred from the calling client to the service and back for each service request. Thus, for multiple service requests a stateful approach can be used based on the proposed transaction server for extended server-based session-management and data persistence. With this *stateful* service behavior also more complex systems are possible with data and parameters remaining on the server for subsequent calls. A complete complex data structure can remain as a central data repository for being queried and modified by different services. This allows overcoming data transfer bottlenecks with larger data volumes as the effectively transferred amount of data between client and server is reduced. It also implies the service based management of an entire generalization workflow. This, however, would then almost lead to a completely server-based generalization system and it assumes that the different services using the stateful data are on one server or are connected via a fast network.

For support services that provide only entity information such as geometries or attributes an extended web based database such as the WFS-T could be used. This transactional web feature server allows not only the data retrieval but also transactional operations including update, insert and delete. Thus, services could retrieve from such a WFS data for the processing and finally save the results back to the server instead of sending it to the client. This scenario would allow generating workflows between different services without having to transfer the data each time to the client. The services are only accessing the data which they need. In a more specialized environment Bobzien et al. (in press) are using a data model with persistently saved information about structural and semantic feature relations with web services (cf. §2.2.2). This environment, however, uses a conventional relational database in a closed intranet to store the data model which can then be accessed by the services using standard SQL.

Developers of generalization algorithms can benefit from the use of support services instead of having to re-implement or integrate source code. However, they still need to write a parser or wrapper routine to access and read the data structure. This is clearly a hurdle that must be taken, but usually applies also for the use of a software library. Since the web services are respecting common interfaces, different support services providing different graph structures, for instance, can then be used without having to recreate the entire parser. A developer can then even rely on the same supporting data structures while using different platforms.

The use of support services in the development of generalization operators and processes might sometimes be too inflexible due to the functional subdivision of the different algorithmic steps. Thus it must be evaluated for every generalization process at which degree of granularity the subdivision into smaller or larger functional components and service calls is appropriate.

3.1.4 Research Paper 4: Automated Processing for Map Generalization with Modular Operator Services

Neun M., D. Burghardt and R. Weibel (submitted). Automated Processing for Map Generalization with Modular Operator Services. Submitted to *GeoInformatica*.

In map generalization various operators are applied onto the different map features in order to maintain and improve the legibility of a map after the scale has been changed. These operators must be applied in the proper sequence and the quality of the results must be continuously evaluated. This paper focuses on the category of the process services (cf. Research Paper 2) allowing control of a generalization process consisting of several operator services with the help of support services. Thus this paper presents an approach for the exploitation of structural knowledge in a generalization process (responding to the thesis objective 3).

Contributions

Important for controlling a generalization process is the formalization of the requirements of the resulting map which can be done with constraints. The constraints describe conditions that have to be met and changes or conflicts that have to be prevented in order to make the map legible. A constraint can have any value between fully satisfied and fully violated (expressed by the range from 0.0 to 1.0). The approaches presented in this paper use constraints for the selection and chaining of different independent generalization operators into the optimal sequence (research question f).

Figure 3.3 shows conflicts that arise during map generalization. The left example shows the conflicts which arise when a larger symbolization for roads is applied. The buildings then overlap with the road symbols (grey lines). Secondly, some of the buildings are too small and would not be visible at the target scale (right example). Therefore different generalization operators must be applied an optimal sequence (in this example building simplification, typification and displacement).

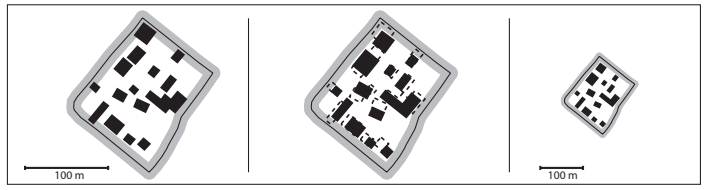


Figure 3.3: Conflict due to road symbolization, solved through simplification, deletion and displacement (examples show the source scale, the generalized result at the source scale and at the final target scale)

Manifold generalization algorithms, available as operator services in the WebGen framework, can be combined to jointly form a powerful workflow. These operators can differ in the severity of the performed generalization task (from small to quite

radical changes in the map) and in the level of sophistication (from unintelligent algorithms up to context aware ones). Examples are operators for simplifying building outlines, for enlarging or shrinking buildings, for displacing buildings, for aggregating buildings, as well as for typifying and removing buildings.

This paper demonstrates how the versatility of a service based architecture can be exploited in generating automated generalization processes in a way that is not possible with more traditional computing paradigms. In particular, the service based approach allows modular generalization processes to be built that can be flexibly extended by introducing additional generalization operators or supporting facilities, that can be coupled to arbitrary optimization strategies. The paper shows a successful use of parallel processing for increasing the performance of the optimization strategies.

The focus of this work is on introducing and revising different constraint based combinatorial optimization techniques for the control of the generalization process using the modularity and flexibility of a services based architecture. Having a relatively large set of different operators available as services the goal is to select the right operator sequence for a given problem. This calls for the use of combinatorial optimization techniques which apply and evaluate the different operators. Due to the use of constraints for continuous evaluation the optimization strategies can be seen as an iterative process between conflict analysis and conflict resolution. The optimization approaches presented in this paper apply the operators onto building partitions derived from a trans-hydro-graph (Timpf, 1998).

Different optimization strategies including *hill climbing*, *simulated annealing* and *genetic deep search* are presented and evaluated experimentally. In the experiments different data sets with scale 1:25'000 are generalized automatically to a scale of 1:50'000. Thereby the optimization strategies showed a substantial difference in terms of amount of conflict reduction and processing performance. The *genetic deep search* shows a significantly higher amount of conflict reduction by evaluating a range of possible operator sequences instead of only one as is done by the other strategies. The long processing time, however, inhibits use with larger datasets or in near real-time environments. It is interesting to note that with the presented processing strategy the *simulated annealing* approach optimizes only slightly better than the *hill climbing* approach while needing significantly more processing time. This finding is in contrast with the results of (Ware et al., 2003); the main reason for this difference is the coarser operator granularity in the here presented approach.

Discussion

The processing strategies used in this paper all depend on the ability to describe the desired cartographical quality sufficiently with constraints. Strictly measurable properties like the size of a building or the distance between two objects can clearly be formalized with constraints. The formalizing of, for example, visually important structural relations of a map however still has many open questions. Some cartographic quality rules are even differently interpreted by different cartographers. The

issue of formalizing constraints is currently being addressed by the EuroSDR project (Burghardt et al., 2007).

The approaches presented in this paper only looked at the generalization of buildings constrained by their surrounding roads. In a real scenario these roads and other map features must also be generalized accordingly. Therefore a combined semi-automatic workflow using predefined workflows in combination with constraint based processing strategies could be used. Not yet answered is the question whether the proposed optimization technique could also be applied to road generalization. Therefore ways are needed to subdivide the road network into smaller partitions that can be treated independently.

The optimization approach presented in this paper currently only evaluates constraints at group level. Thus the approach is controlled by the satisfaction of the constraints of all features in one partition. This satisfaction is calculated by a cost function where the individual constraints are combined with different weights. Such a cost function, however, can suppress small but important effects. Specific constraint violations of one single feature are therefore in the responsibility of the operators as they are usually trying to solve the most important constraint violations first.

With a set of operators there exists a large space of possible operator sequences which can be conducted interactively. The aim is to find an optimal result (minimum) from this global search space of possible solutions. A problem which is not treated in the presented approaches is the fact that the minima are probably not equally distributed over the search space. Thus, with more knowledge about the used operators and about prevailing optimal sequences, a pre-selection of the search space could be made in order to speed up the processing time. This however conflicts with the targeted use of independent operators that are not known beforehand in a generic way.

The different optimization strategies like *hill climbing*, *simulated annealing* and *genetic deep search* have quite different outcomes in terms of amount of conflict reduction and in terms of processing performance. The *genetic deep search* proved to be usable as a reference for the results of the two other approaches. It tries to approach a full exhaustive search and thus to test as many as possible different sequences instead of only one with the *hill climbing* or *simulated annealing* approaches. A real full exhaustive search is clearly not possible due to the increasing complexity. Currently the combination of the *genetic deep search* approach with learning techniques is being investigated. First results are presented in Research Paper 5 included in this thesis (see §3.1.5) in order to give an outlook for future developments.

The use of parallel processing techniques proved to be useful for increasing the process performance. This approach could perhaps even be extended to take advantage of real grid computing with a large number of distributed services, with techniques to control the distribution of tasks and with load balancing. It could also be possible to use algorithms that are only available for special computer architectures or parallel processing systems in such a grid if they were available through a web service interface. For an advanced utilization of distributed services and of grid computation ways have to be found for separating and distributing generaliza-

tion tasks. Therefore independently processable problems must be identified and advanced methods for the subdivision of the map are required.

3.1.5 Research Paper 5: Automated Sequencing of Generalisation Services based on Collaborative Filtering

Burghardt D. and M. Neun (2006). Automated sequencing of generalisation services based on collaborative filtering. In: M. Raubal, H. J. Miller, A. U. Frank and M. Goodchild (eds): *Geographic Information Science. 4th Inernational. Conference on Geographical Information Science (GIScience 2006)*, IfGIprints 28, pp. 41-46.

Research Paper 4 introduced and discussed processing approaches for the automated selection and sequencing of generalization operators. Learning techniques can be used to assist and accelerate these selection and sequencing processes. This paper presents the use of a continuous learning knowledge base in conjunction with collaborative filtering (Linden et al., 2003) for the automated prediction of operator sequences. With this paper first conceptual and experimental results were presented at the GIScience conference in 2006. The work is still in progress on and thus, since this paper is the last one in this thesis, it also gives an outlook on prospective developments in the domain of processing strategies.

Contributions

Map features or groups of map features can be placed inside a constraint space (Burghardt and Steiniger, 2005) depending on their cartographic properties. The distance of the features from the origin is a measure of cartographic conflicts.

The automated generalization process is executed in iterations chaining appropriate generalization operators (cf. figure 3.4). With every step of the iteration different available operators are applied and tested with the current generalization situation, representing a map feature or a group of map features. The signatures of the constraint violations (constraint pattern showing the position in the constraint space) before each step are saved together with information about the successfully applied operator sequences respective parameter settings in a knowledge base. This information represents enriching information used for the control of the generalization process (research question g).

The knowledge base will be built up in a continuous way. At the beginning it contains only one entity, which recommends all available generalization operators equally. Thus, initially all available operators are applied and evaluated. With a growing knowledge base fewer and fewer operators must be applied since the most probable operator sequences, stored in the knowledge base, can be predicted (cf. figure 3.4) by matching to similar constraint patterns. Thus, the most promising operators with their parameter settings are applied to the generalization situation. The prediction quality for the best suited operator sequences depends on the number of entities in the knowledge base which is growing with every execution step. A

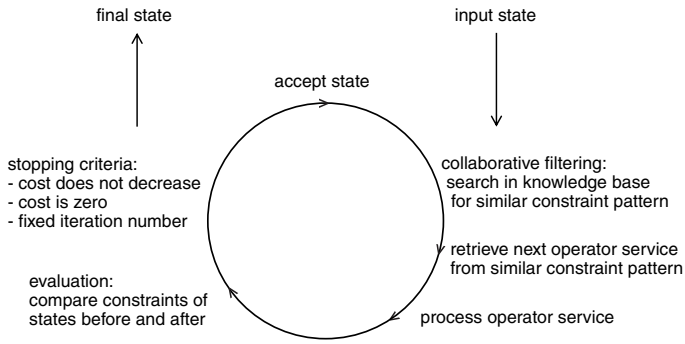


Figure 3.4: Flow chart of the applied iterative generalization process

collaborative filtering technique (Linden et al., 2003) guaranties a fast extraction of operator sequences from a large knowledge base. This technique is also used for example in e-commerce web sites for making customer recommendations.

The approach works for generalization situations with single features as well as with groups of features. The examples presented in this paper use groups of features as processing entities, each representing a partition of buildings, derived automatically from a street and river network. Thus, the operators are always applied to such a building partition and also the constraint violation signatures are evaluated partition-wise.

Discussion

The optimization technique presented in this paper uses a modified hill climbing approach for the iterative sequencing of operators. Thus, the continuous learning knowledge base can only predict operator sequences that reduce constraint violations with every processing step. Sequences that allow the execution of an operator that creates temporary constraint violations followed by an improvement by another operator are not possible. An example would be the deletion of some buildings in order to allow the successful displacement of the remaining buildings. This sequence is sometimes not possible as the deletion can even cause more constraint violations. A possible solution for this problem would be a training phase where an exhaustive testing of many different sequences is applied as in the genetic deep search approach presented in Research Paper 4. Another possibility would be the use of combined "virtual" operators of length 2 or 3. These virtual operators are predefined sequences which are sometimes unlikely to be selected in a hill climbing approach such as combinations with an initial deletion of buildings and a subsequent displacement.

Currently the knowledge base is implemented as a simple Java service which keeps the data in the main memory while being active. This ensures quick response times for the collaborative filtering. In advanced processing systems possibly millions of entries in such a knowledge base, describing every possible situation, are imaginable. Therefore it should be possible to implement the knowledge base as

an application in a relational database management system (RDBMS) using e.g. stored procedures for the collaborative filtering. This application could then be made available as a service within the WebGen framework, even allowing the use of one knowledge base by different processing services.

The first experiments have shown that constraint violation signatures can be used to predict operator sequences. These predictions depend on the ability to describe the violations optimally and to differentiate the different situations sufficiently. In order to better understand these dependencies advanced visualizations of constraint violations in similar situations (selected for example by experts) as well as visualizations of the entries in the knowledge base might help.

3.2 General Discussion

In the five Research Papers the concept of generalization services was first motivated (Research Paper 1) and then proposed and implemented (Research Paper 2). Following that the applicability of the services concept was illustrated by the large family of support services which serve for the provision of data enrichment methods (Research Paper 3). Finally the usage of these support services for the control of the generalization process was implemented and evaluated (Research Papers 4 and 5).

This section attempts a synthesis of the discussions of the five individual Research Papers, highlighting the strengths as well as the weaknesses and open issues of the presented research.

3.2.1 Strengths

With the proposed WebGen framework various functionalities can be provided as services offering a large interactive generalization toolbox locally in an organization as well as over the internet. Such tools are for example data enrichment functions such as measures from spatial analysis or complex neighborhood graphs provided as support services. Probably the most important and salient services are the operator services which provide the generalization operators. Finally, even complex generalization functionalities can be offered through the so-called process services.

The use of services has also been demonstrated in the GiMoDig project (Sarjakoski et al., 2005), although restricted to a real-time middleware approach using basic generalization tools for the provision of mobile devices with map data from different sources. The WebGen framework, representing the prototype of interactive generalization services, has in the meantime influenced a number of other developments (Bobzien et al., 2006; Foerster and Stoter, 2006; Regnauld, 2007). Thus the concept of services is regarded as a promising approach.

The differentiation into service categories (support, operator, process) allows the functional subdivision of complete generalization processes. This allows the use and sharing of generalization functionalities on very different levels of granularity. Whole generalization service processes can be used or tested. On the other hand

for example individual support services can be re-used for the prototyping of new generalization ideas. Advanced data enrichment methods are no longer limited to one specific research group or platform but can be provided as support services to the community at large (see Research Paper 3). New algorithms can be provided for the testing by others without loosing control over the source code. Different independent generalization operators can be combined in one automated and self-evaluating generalization process (see Research Paper 4). All these possibilities may boost the research for automated generalization (Regnauld, 2006).

The following list gives an overview of the strengths and benefits of the services based approach presented in this thesis:

- The proposed WebGen framework implements an interactive toolbox of generalization services which can be used stand-alone as well as by other services to build complex workflows. It eases the publishing of and access to generalization functionalities, including algorithms or data enrichment methods.
- The functional subdivision of the generalization process into support, operator and process services offers many ways to use and also to re-use generalization functionalities. Only missing functionalities have to be implemented as the existing tools are available as services even from different platforms. This will boost research especially in the prototyping phase.
- Furthermore complex spatial data structures (triangulations, graphs, Voronoi diagrams, etc.) can be provided through services. Since these data structures are heavy to compute, the stateful service approach is advocated for re-use and updating.
- Services can be used for creating generalization processes of loosely coupled independent support and operator services. This allows the use of constraints and optimization techniques in the services environment.
- Services can be registered and described in a central registry allowing the dynamic retrieval of services on different distributed systems. The generalization services can be described in a generic way using the interface descriptions.
- Generalization services are available to be used and tested independently of language bindings in the user's own environment. Plug-ins for new environments can be developed using internet standards, as demonstrated with the development of a plug-in for the Clarity system (Regnauld, 2007).
- Generalization services can be used for the coupling of heterogeneous systems in research and possibly also in map production. Generalization functionalities can be shared without users needing physical access to the server performing the computation.
- Generalization services can be used by other geo-spatial web services for performing generalization tasks. The services can also be used in a middleware scenario.

- Users don't have to know the implementation source code of a service in order to use it. Thus the owners remain in control of the code and can secure intellectual property rights. Co-operations between commercial and non-commercial parties are thus more feasible.
- The services architecture allows also the utilization of parallelization techniques. Even inside a workflow services can be executed in parallel on one or multiple servers.

3.2.2 Weaknesses and Open Problems

The presented methods do not only have benefits. Some limitations and unsolved issues must also be mentioned:

- It is obvious that large amounts of data as well as high frequencies of service calls may provoke problems with the transfer times and network latencies. For researchers these constraints are probably less important than in a real production scenario.
- Client and server plug-ins must be implemented for each platform from which services are accessed or provided. The plug-in also has to provide a translation interface between the local data format and the standard data format used by the service. The development of such a plug-in is a one-off cost. But sometimes the use of a software library might be faster to implement even if the plug-in is more sustainable.
- In some cases the high granularity of a strict functional subdivision into multiple services might not be appropriate and developers might prefer the use of a traditional software library with geo-spatial functions. On the other hand, it is also possible that generalization tools are "packaged" as services at a more integrated, more coarse-grained level. The high granularity of service packaging is not a must.
- The interface descriptions of the WebGen framework have only a limited, but for many applications sufficient, amount of meta-information about the data and parameters involved. For achieving full interoperability between systems and knowledge domains the data and parameter semantics must be taken into consideration (see also §4.3.1).
- The algorithms and services have to follow some standards and conventions of the service interfaces. Thus developers have no 100% liberty in designing the interfaces and the interactions with the service users.
- Various services are offering generalization functionalities but the descriptions are so far only textual and thus only human-readable. Therefore the capabilities of services and the dependencies from other services must be described in a generic way respecting the service semantics. As a starting point meta-data with a classification of the service characteristics could be used. More about this topic in section 4.3.1.

- The services concept expects that data is processed on servers provided by other organizations. This might pose problems with data copyrights and security for securing the intellectual property. The service providers could intercept the data transfers and get hold of protected data. Therefore digital rights management would be required in order to use our services framework operationally between multiple organizations.

- The success of the services platform is depending on the adoption by researchers and national mapping agencies. This adoption, however, requires an initial effort for the implementation and integration of the plug-ins. If there are no plug-ins for commonly used GIS and mapping systems such as ESRI's ArcGIS or Intergraph's GeoMedia, the adoption of the services platform will be severely limited.

Chapter 4

Conclusion and Outlook

This thesis presented research about methods for the automation of map generalization. The main goal was to support the automated generalization of maps and map data with auxiliary enriching information. Therefore the thesis addressed the following three objectives:

1. Storage and provision of enriched data using ***generalization services***

2. Identification and formalization of structural relationships for ***data enrichment***

3. Exploitation of enriched data for ***adaptive generalization***

These three objectives were mainly pursued in the Research Papers 2, 3 and 4 whereas the Research Papers 1 and 5 served as motivation and outlook, respectively, for the research carried out in this thesis. Objective 1 is mainly accomplished in Research Paper 2 by introducing and describing the WebGen framework for generalization services. Research Paper 1 gives the underlying motivation therefore and the remaining Research Papers demonstrate application scenarios of generalization services. Objective 2 is accomplished in Research Paper 3 by applying the concept of support services for providing data enrichment, by developing a categorization of possible support services and by reviewing structural knowledge types used in generalization with a special regard onto relations between map objects. Objective 3 is addressed in the Research Papers 4 and 5 by developing processing services to control a generalization process of various generalization operator services with the help of enriching information provided by support services.

Section 4.1 is recalling these objectives in order to conclude this thesis with a summary of the main contributions. Following that the research questions which were posed in the introduction (§1.2.1) are systematically answered in section 4.2 and finally section 4.3 provides an extended outlook discussing possible challenges and avenues for future research.

4.1 Main Contributions

The main contribution of this thesis is the use of a services oriented architecture for an interoperable generalization research platform. Thereby a conceptual framework including three service categories (support, operator, process service) was developed and the feasibility of the approach was proven with the implementation of the WebGen framework providing a comprehensive solution for a common interoperable generalization platform. Its relative simplicity, language neutrality, and ease of publishing and accessing algorithms facilitate joint research and the coupling of different generalization platforms. This framework allows the loose coupling of different generalization algorithms and other supporting methods for use in complex generalization workflows. The three service categories influenced the overall structure and ordering of the research topics and thus provide a good outline for describing the other contributions of this thesis:

Support Services are enriching the raw input data with additional information or expressing structural and spatial relationships. This data enrichment is demanded by the objectives 1 and 2. The research for objective 2 derived from the insight that many methods for the enriching of spatial data with structural knowledge are well known or even implemented but they are often implicitly contained in the specific algorithms or use proprietary data models and formats. In contrast, our support services allow the provision of the enriching data explicitly and in an interoperable way. Furthermore, many of these supporting data structures are expensive or difficult to calculate. The approach of a stateful support service allows the re-use and modification of once generated data structures by different operator and process services. Developers of generalization algorithms can now use the support services in conjunction with generalization operator services without having to know in detail how the support data is generated. Thus the prototyping of new generalization algorithms and processes is facilitated using existing services in conjunction with newly developed routines.

Operator Services deliver the functionalities of individual generalization operators and take enriched data as input. Due to the service architecture of the WebGen framework various operators from different sources can be integrated and used. A large set of generalization operators (presently more than 15, see subset in Research Paper 4) are available as services. These operator services can be accessed and used by everybody using a WebGen plug-in and they can also be called by other services such as the process services.

Process Services use the two other service types in order to control the generalization process as a workflow. With the process services it was shown how enriched data provided by support services can be used to control a generalization process of several independent generalization operators. The presented building generalization approach uses combinatorial optimization techniques for the chaining of generalization operators provided as independent services. The evaluation of the optimization

results is accomplished using cartographic constraints calculated by support services. The selection of the operators was done using different heuristics. These showed substantially different results in terms of processing time and quality of the result. Additionally learning techniques were applied for increasing the performance and quality. This was achieved by using a knowledge base which stores operator sequences together with the constraint situations they resolved. Finally, with the process services a successful use of parallel computing techniques for generalization was also shown.

4.2 Answering the Research Questions

In the conclusions made above as well as in the synthesis of the Research Papers (§3.1) the main links between the objectives of this thesis, the research questions and the Research Papers were already mentioned. This section attempts to systematically answer the different research questions (§1.2.1) drawing from the results brought about by the Research Papers.

a) *What are the requirements for an open research and development platform?*

The first requirement for an open generalization system is the sharing of generalization algorithms and data enrichment methods for research. Secondly, offering generalization functionalities externally to the growing number of geospatial services is also highly desirable. Therefore a services based architecture is most promising for an open platform. Research Paper 1 motivates and discusses these requirements in the context of open standards, open architectures and open source developments for geographic information. As a possible starting point the development of small generalization services with narrowly defined functionality was identified.

b) *Can a services based architecture be used for the sharing of algorithms and enriching information in an interoperable way?*

The WebGen framework proposed in Research Paper 2 evolved from a prototype for the sharing of simple generalization functionalities. It shows a complete framework that makes generalization algorithms as well as supporting methods and data structures available . Therefore three main categories of generalization services (support, operator, process) were identified. These service categories allow a complete functional decomposition of a generalization process. The services themselves can be offered externally in order to allow the sharing of functionalities as well as the use by other geospatial services.

c) *What types of structural knowledge can be provided by services for the support of generalization methods?*

The structural knowledge can be provided by the so-called support services. Research Paper 3 delivers a comprehensive taxonomy of generalization support services. The taxonomy differentiates enriching entities, consisting of simple attributes or geometries, and enriching relations, which express the various hierarchical and non-hierarchical relations between map objects. The attributes or geometries can be directly used by generalization operators as they are represented in standard formats. The relations that involve complex data structures such as graphs, however, require advanced data formats for being usable by generalization operators.

d) *How is the knowledge about map object relations being generated and used?*

Many methods for the enrichment of spatial data with structural knowledge, such as the relations between map objects, are well known or even already implemented. These methods, however, are often only implicitly contained in algorithms or do

require proprietary data formats. Research Paper 2 reviews the different structural knowledge types with a special regard to the knowledge about the large family of relations ranging from directed or undirected graphs, such as transport graphs and triangulations, to hierarchies which can be represented by trees.

e) *Can this knowledge about relations serve as enriching information?*

The knowledge about relations can be provided by support services. In Research Paper 4 the utilization of these support services is illustrated on four implementation examples of support services being used by generalization operators. These examples demonstrate possible usage scenarios and benefits of using support services. The first example is a service for the provision of a triangulation and a proximity graph which is used for the building typification and displacement. Other examples are support services for the detection of building alignments, for the re-classification and for the detection of urban structures.

f) *How can an adaptive generalization process be modeled using a framework of generalization services?*

Adaptive generalization seeks to select the right operators in the right sequence for a given map situation and parameterize these operators accordingly (see §1.1.3). Within the WebGen framework a large set of independent generalization operators are available as services for use in an adaptive generalization process. We show in Research Paper 4 that it is possible to use a combinatorial optimization approach for the selection and chaining of these independent operator services. Different heuristics can be used for the operator selection in the optimization approach. Thereby cartographic constraints are used for controlling and evaluating the cartographic quality continuously.

g) *How can the enriched data be exploited in order to control an adaptive generalization process?*

The optimization approach presented in Research Paper 4 uses cartographic constraints for evaluating the cartographic quality of generalization solutions. These constraints are calculated and updated by support services providing the constraint violations as attributes of the map objects. Thus these support services provide constraint violation values as enriching information. Research Paper 5 extends the process service approach of Research Paper 4 by using a knowledge base of previously executed successful operator sequences. This knowledge base relates the successful operator sequences with the cartographic constraint violations which they solved. Thus this knowledge base contains also enriching information which is used for controlling and speeding up the generalization process.

4.3 Ongoing Challenges and Outlook

With the advent of the WWW new opportunities were created for map use and exchange of spatial data. Web based applications such as Google Earth / Google Maps or Windows Live Local are now commonly used. Mashups allow also the non-expert user to combine topographic information and aerial or satellite imagery even with own or custom made data. Car navigation systems have become a commodity and allow the presentation of cartographic information in spoken form as well as with 3-dimensional views. Thus, maps nowadays are a read/write medium instead of a read-only medium (Dodson, 2005).

The growing use of geographical information systems and with this the widespread need of geographic data also increased the demand for automated generalization. In this context, new tasks of information filtering and abstraction arise (Jones et al., 2002) in order to adapt maps or geographic data to the various activities and needs. Mapping agencies have to provide more and more digital data and services instead of traditional paper maps. This implies also that users expect to use this data for own tasks with their own software. Therefore there is a growing need for approaches to make data technically and semantically usable in an interoperable way. Also mobile applications such as location based services and navigation systems are requiring more and more tailor-made cartographic information. Such adaptive maps conform autonomously the map content and the visualization to the information required by the user (Reichenbacher, 2004). This increasing demand for tailor-made cartographic data strengthens the requirement for a fully automated comprehensive generalization system.

As an outlook of this thesis this section highlights some work that is currently starting off and which is related to the topics of automated generalization, interoperability and web services (§4.3.1). To conclude this thesis further improvements and future challenges are discussed which arose during the research for this thesis (§4.3.2).

4.3.1 Building a Comprehensive Generalization System

In this thesis methods and a framework for generalization research have been presented. The objective of data enrichment for the improvement of generalization operators and for the control of generalization processes gave the initial motivation for this research. With the use of the generalization services model the interoperable utilization of different data enrichment methods (support services) and of different generalization algorithms (operator services) could be demonstrated. Furthermore first results for the automated control of a generalization workflow as well as methods for increasing the performance of generalization were presented (process services). At this stage we are technically in a position to establish a comprehensive and open generalization system using a services architecture.

The use of services, as proposed in this thesis, allows the combined use of different tools and algorithms from different platforms. Currently a human controller is needed at least for selecting a set of operators which must be applied and for

translating the desired cartographic specifications into rules or constraints for the automation. In order to allow the fully automated orchestration of different services more information about the capabilities of each service is required. The capabilities of such a service include also the information about the granularity of the service and its interdependencies with other services. This information must be detailed and generic and it must be understandable by the computer such that no further expert knowledge is required for guiding the generalization process. This would allow the system to be also utilized by cartographic laypersons. A solution for representing this information could be an ontology containing service capabilities and dependencies. A desirable scenario would be for example that a user can say "the line is too detailed" instead of having to tell the system to use a simplification and a smoothing algorithm and then to specify the right parameters.

A generic generalization system relies on the ability to combine heterogeneous input data sources with various different generalization services in order to turn them into the desired user product. Regnauld (2007) proposes a first sketch of such an open architecture to provide on demand mapping. This architecture would allow the British Ordnance Survey as a map producer to use generalization tools from different sources in a generic generalization system. The architecture is illustrated in figure 4.1. This proposal extends the concept of generalization services in the WebGen framework with the use of different ontologies to overcome the need for semantical interoperability which was also already mentioned in the discussion of this thesis. The generic generalization system in this proposal would combine the different ontologies, analyze the user requirements and select the data and the required generalization services accordingly.

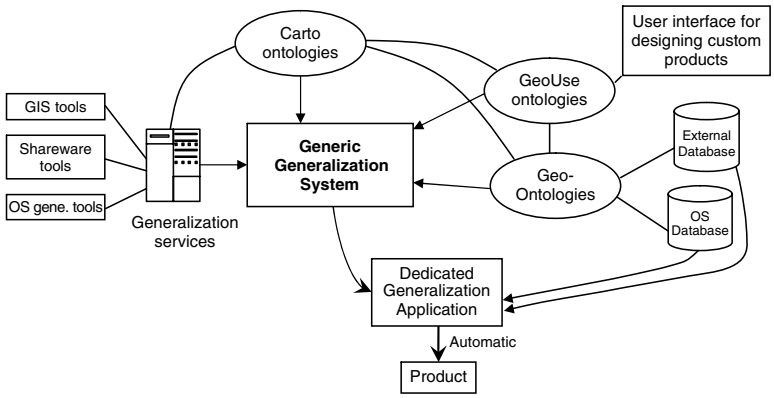


Figure 4.1: System architecture for tailoring on-demand products (adapted from Regnauld (2007): Copyright Ordnance Survey, UK)

The system architecture in figure 4.1 relies on the semantical definitions of three ontologies. These ontologies are basically acting as translators between the heterogeneous inputs (services, data sets and user requirements) and the generalization process logics (in the generic generalization system). The Geo-Ontologies relate the geographic and topographic domain (Goodwin, 2005) of the internal data semantics

with the different domains of external data sources. The GeoUse-Ontologies relate usage scenarios and user requirements with the complex generalization processes. The Carto-Ontologies describe cartographic rules and relate them to generalization operator capabilities such that the generic generalization system is not product dependent. These three ontologies must be interrelated in order to share the common concepts of the internal system.

Thus one of the major current and future challenges is the development of these three types of ontologies in order to allow the semantic interoperability and thus the generic automation of the generalization process in services oriented architectures such as the WebGen framework. With regard to map generalization, the need for such generic and machine readable algorithms or service capability descriptions has already been mentioned by several other researchers at a meeting in Dagstuhl[1] in spring 2006. With regard to more general distributed geo-services Lemmens (2006) proposes a first approach to formally describe services based on semantics.

In Research Paper 4 and 5 we demonstrated the use of constraint violation signatures for triggering generalization operator sequences. The before-and-after comparison of the signatures that have been calibrated against standard datasets would return values on how the different constraints are influenced by a particular operator. This differential signature of the operator describes its capabilities simply by the results and thus provides a generic and self-configuring mechanism for the description and characterization of each generalization service. This service description can be created automatically and then used automatically in different generalization processes. While they are difficult to read for humans such descriptions are very well machine readable. Thus they are applicable especially in automated constraint based chaining and optimization techniques. For a broad generic use these constraint signature descriptions could also be combined with an ontology classifying the general service capabilities.

4.3.2 Further Challenges and Improvements

Data Partitioning

Generalization techniques are often very complex as they have to compute complex data structures representing for example the neighborhood relations. The computational complexity of the algorithms is often worse then linear in relation to the number of map objects and the execution time may grow excessively. Thus data sets with too many single features are difficult to generalize at once. On the other hand many map producers are moving towards storing spatial databases of entire countries in a seamless manner. In order to enable the automated generalization of such large databases strategies are needed for subdividing the data in manageable junks.

[1]Discussion after the talk of Dirk Burghardt about “Generalisation Services on the Web” at the Dagstuhl workshop on “Spatial Data: Mining, Processing and Communicating”, March 2006

The traditional approach known form manual cartography of subdividing the map data into smaller rectangular tiles (e.g. map sheets) is a solution which causes boundary effects at the edges of the rectangles, as these will not normally follow 'natural' borders. Thus other ways of splitting up the map space into smaller independent units called partitions are required, which are not causing boundary effects. For automated building generalization processing services (Research Papers 4 and 5) building blocks, derived from a network of transportation and hydrography lines (also termed trans-hydro graph), were used as smallest independent units. These partitions can be generalized separately as the buildings inside one partition are not influenced across partition boundaries. This approach of using roads and rivers as separating elements is limited and only applicable to objects such as buildings and only if no strong displacement of map objects is required. Thus new partitioning strategies are needed in order to apply the proposed processing approaches onto other map object classes.

Distributed and Parallel Computing

Web service technologies allow new possibilities for the sharing of algorithms and the coupling of platforms but also for the application of parallel and distributed computing in order to boost the performance. Generalization algorithms as well as many other spatial algorithms have at the moment usually a sequential, step-by-step behavior. This works well for smaller datasets. A fully automated generalization system, however, should be able to deal with potentially huge datasets. Therefore the data must not only be split up into smaller partitions but the algorithms and processes must also be functionally re-organized and split into independent operating tasks. Once data partitions and tasks have been separated parallelization can be used for improving the execution performance.

Until very recently, parallel processing was limited to special computers and therefore not accessible to everyone. Therefore previous attempts to use parallel computing in GIS were limited to special cases. With the advent of more and more inexpensive multi-core computers even regular computer users have a "super computer" with multiple processors on their desk. Having available these powerful computers and having the possibility to create clusters of multiple computers it is now a good moment for investigating the use of parallel processing in generalization. First test with a parallel multi-threaded system as shown in Research Paper 4 showed a great potential for being investigated further.

Service oriented environments like the WebGen framework do no rely on the execution on a single computer. Multiple servers can coexist, concurrently offering their computing power. So, besides making the processes multi-threaded it is even possible to employ a cluster of servers and to distribute the work load between them. Another option to explore might be the use of services such as the Amazon EC2[2] or the Sun Grid Compute Utility[3] which provide resizable computing power over the

[2]Amazon Elastic Compute Cloud (Amazon EC2), http://aws.amazon.com/ec2
[3]Sun Grid Compute Utility, http://www.sun.com/service/sungrid/

Internet. Such services offer pure processing power in a distributed environment on a pay-per-use basis without having to buy and maintain a whole cluster of servers. These offerings will also allow researchers or small companies to benefit from high-performance computing at a reasonably low cost.

Final Thoughts

During this thesis some further challenges or desirable improvements did arise. The WebGen framework and most of the generalization services were implemented in Java using the JTS Topology Suite[4] for working with geometry data and the JUMP Unified Mapping Platform[5] as a client and data viewer. JUMP uses an object oriented data model for representing the working data. A problem arises when working with very large data-sets as all the data is kept in main memory at all times no matter whether the data is loaded from a file or a database. Thus a very useful extension for this software would be a technique which allows loading a large dataset without having it completely in memory. A solution, as already demonstrated in van Oosterom and Laffra (1990) and Vijlbrief and van Oosterom (1992), could be the use of a spatial relational database system such as Oracle[6] or PostGIS[7] for keeping the data while only subsets that are generated as a database view[8] are displayed and thus loaded by JUMP in the main memory. A further more advanced option would be the use of a real object oriented database which would store the whole data model of JUMP persistently.

Another prospective development are multi-representation or multi-resolution data structures as well as variable-scale structures. In Neun and Steiniger (2005) horizontal and vertical relations were identified for data enrichment in support of generalization. The project SerAx (Bobzien et al., ress) developed a comprehensive multi-resolution database (MRDB) which implements these horizontal and vertical relations and stores them persistently. As a future challenge such a data structure could be made available as web service such that it could be used by different generalization services. This could possibly be done in the form of a transaction server as proposed in Research Paper 3. Also variable-scale structures such as the GAP-tree (van Oosterom, 2005) could possibly be made available as services with this approach.

For the future of the WebGen framework or other approaches such as WPS not only technical limitations like the overhead due to the XML conversion or the required transfer times are playing an important role. Copyright and security issues might be even more important. The transfer and conversion problems will probably diminish with ongoing advances in computing and network speed. Copyright and security issues, on the other hand, play an important role if national mapping agencies

[4] JTS Topology Suite, http://www.vividsolutions.com/
[5] JUMP Unified Mapping Platform, http://www.jump-project.org
[6] Oracle Spatial, http://www.oracle.com/technology/products/spatial/
[7] PostGIS is the spatial extension for PostgreSQL, http://postgis.refractions.net/
[8] A VIEW in a relational database represents a virtual subset of a database table which dynamically updates if the table or other conditions change.

or other security sensitive parties want to use such an open platform as well. Thus future research and development will have to address issues like the watermarking of geographical data for assuring the copyright as well as the encryption of the web service communications and parameters. Encryption would be particularly useful if even a service provider can not read the data which is processed on their services such that service subscribers could also use confidential data in such an environment.

Finally, the WebGen framework was intended as a proof of concept to show the application of a web services architecture as an open research platform. In order to encourage the adoption and use by researchers and national mapping agencies the development of more plug-ins for servers and especially for clients is required for the different GIS and generalization platforms. Then researchers and developers can use this framework for testing and comparing algorithms as well as for coupling different platforms and for establishing complex process workflows. At the same time the use of such services by a wider GIS community could further the concept of services oriented architectures for GIS applications.

Bibliography

1Spatial (2007). Radius Clarity. http://www.1spatial.com (accessed 03/2007).

Albrecht, J. H. (1999). Universal analytical GIS operations - a task-oriented systematization of data-structure-independent GIS functionality. In Craglia, M. and Onsrud, H., editors, *Geographic Information Research - Trans-Atlantic Perspectives*, pages 577–591. Taylor & Francis, London.

Andrienko, N. and Andrienko, G. (1999). Interactive maps for visual data exploration. *International Journal of Geographic Information Systems*, 13(4):355–374.

Apache (2007). The apache tomcat servlet container. http://tomcat.apache.org (accessed 07/2006).

Armstrong, M. P. (1991). Knowledge classification and organisation. In Buttenfield, B. and McMaster, R., editors, *Map generalization: Making Rules for Knowledge Representation*, pages 86–102. Longman, London.

Axes Systems (2007). Automated gis cartography solutions. http://www.axes-systems.com (accessed 04/2007).

Badard, T. and Braun, A. (2003). OXYGENE: An open framework for the deployment of geographic web services. In *Proceedings of the 21st International Cartographic Conference*, Durban, South Afrika.

Bader, M., Barrault, M., and Weibel, R. (2005). Building displacement over a ductile truss. *International Journal of Geographical Information Science*, 19(8-9):915–936.

Bader, M. and Weibel, R. (1997). Detecting and resolving size and proximity conflicts in the generalization of polygonal maps. In *Proceedings of the 16th ICA/ACI Conference*, pages 1525–1532, Stockholm.

Bard, S. (2004). Quality assessment of cartographic generalisation. *Transactions in GIS*, 8(1):63–81.

Barrault, M., Regnauld, N., Duchêne, C., Haire, K., Baeijs, C., Demazeau, Y., Hardy, P., Mackaness, W., Ruas, A., and Weibel, R. (2001). Integrating multi-agent, object-oriented and algorithmic techniques for improved automated map generalization. In *Proceedings of the 19th ICA/ACI Conference*, pages 2110–2116, Beijing.

Beard, K. (1988). *Multiple representations from a detailed database: a scheme for automated generalization*. PhD thesis, University of Wisconsin-Madison.

Beard, K. (1991). Constraints on rule formation. In Buttenfield, B. and McMaster, R., editors, *Map Generalization: Making Rules for Knowledge Representation*, pages 121–135. Longman, London.

Bistarelli, S., Fargier, H., Montanary, U., Rossi, F., Schiech, T., and Verfaillillie, G. (1999). Semringbased csps and valued csps: Frameworks, properties, and comparison. *Constraints: An international journal*, 4(3).

Bobrich, J. (1996). *Ein neuer Ansatz zur kartographischen Verdrängung auf der Grundlage eines mechanischen Federmodells*. PhD thesis, Deutsche Geodätische Kommission, München, Reihe C, H. 455.

Bobrich, J. (2001). Cartographic map generalization in urban districts. In *Proceedings of the GIS Research UK, 9th Annual Conference*, pages 513–515.

Bobzien, M., Burghardt, D., Petzold, I., Neun, M., and Weibel, R. (in press). Multi-representation databases with explicitly modelled intra-resolution, inter-resolution and update relations. *Cartography and Geographic Information Science*.

Bobzien, M., Burghardt, D., Petzold, I., and Weibel, R. (2006). Ableitung von digitalen vektormodellen ergebnisse des projektes drive. *Kartographische Nachrichten*, 56(5):254–262.

Brassel, K. and Weibel, R. (1988). A review and conceptual framework of automated map generalization. *International Journal of Geographic Information Systems*, 2(3):229–244.

Brazile, F. (2000). *Semantic Infrastructure and Methods to Support Quality Evaluation in Cartographic Generalization*. PhD thesis, Department of Geography, University of Zurich.

Burghardt, D., Bobzien, M., Petzold, I., and Weibel, R. (2005b). Cartographic generalisation of large scale maps with axpand. In *of International Symposium on Generalization of Information, ISGI*, pages 147–159, Berlin.

Burghardt, D. and Cecconi, A. (2007). Mesh simplification for building typification. *International Journal of Geographical Information Science*, 21(3):283–298.

Burghardt, D. and Meier, S. (1997). Cartographic displacement using the snakes concept. In Förstner, W. and Plümer, L., editors, *Semantic Modeling for the Acquisition of Topographic Information from Images and Maps*, pages 59–71. Birkhäuser Verlag, Basel.

Burghardt, D. and Neun, M. (2006). Automated sequencing of generalisation services based on collaborative filtering. In Raubal, M., Miller, H. J., Frank, A. U., and Goodchild, M., editors, *Geographic Information Science*, 4th International Conference on Geographical Information Science (GIScience 2006), IfGIprints 28, pages pp. 41–46.

Burghardt, D., Neun, M., and Weibel, R. (2005a). Generalization services on the web -– a classification and an initial prototype implementation. In *Proceedings American Congress on Surveying and Mapping, Auto-Carto 2005*, Las Vegas, USA.

Burghardt, D., Neun, M., and Weibel, R. (2005b). Generalization services on the web -– a classification and an initial prototype implementation. *Cartography and Geographic Information Science*, 32(4):257–268.

Burghardt, D., Schmid, S., and Stoter, J. (2007). Formalisation of cartographic requirements by constraints specification. In *10th ICA Workshop on Generalisation and Multiple Representation*, Moscow, Russia. http://ica.ign.fr/ (accessed 06/2007).

Burghardt, D. and Steiniger, S. (2005). Usage of principal component analysis in the process of automated generalisation. In *XXII International Cartographic Conference*, A Coruna.

Buttenfield, B. P. and McMaster, R. B., editors (1991). *Map Generalization: Making Rules for Knowledge Representation*. Longman, London.

Cartwright, W., Miller, S., and Pettit, C. (2004). Geographical visualization: Past, present and future development. *Journal of Spatial Science*, 49(1).

Cecconi, A. (2003). *Integration of Cartographic Generalization and Multi-Scale Databases for Enhanced Web Mapping*. PhD thesis, Department of Geography, University of Zurich.

Cecconi, A. and Galanda, M. (2002). Adaptive zooming in web cartography. *Computer Graphics Forum*, 21(4):787–799.

Channabasavaiah, K., Holley, K., and Tuggle, E. (2003). Migrating to a service-oriented architecture. IBM DeveloperWorks. http://www-128.ibm.com/developerworks/library/ws-migratesoa/ (accessed 09/2006).

Christophe, S. and Ruas, A. (2002). Detecting building structures for generalisation purposes. In Richardson, D. and van Oosterom, P., editors, *Advances in Spatial Data Handling*, pages 419–432. 10th international Symposium on Spatial Data Handling, Ottawa.

Conrad, S. (1997). *Föderierte Datenbanksysteme*. Springer, Berlin.

Cromley, R. G. and Campbell, G. M. (1992). Noninferior bandwidth line simplification: Algorithm and structural analysis. . *Geographical Analysis*, 23(1):25–38.

Crompvoets, J. (2006). *National Spatial Data Clearinghouses: Worldwide development and impact*. PhD thesis, Wageningen University, Netherlands.

Curbera, F., Nagy, W., and Weerawarana, S. (2001). Web services: Why and how. http://researchweb.watson.ibm.com/people/b/bth/OOWS2001/nagy.pdf (accessed 04/2007).

Cuthbert, A. (1998). User interaction with geospatial data. OpenGIS® Project Document 98-060, http://www.opengeospatial.org/specs/ (accessed 07/2006).

DeLucia, A. A. and Black, R. T. (1987). A comprehensice approach to automatic feature generalization. *Proceedings of the International Cartographic Conference, Morelia, Mexico*, pages 169–192.

Dodson, S. (2005). Get mapping – as mapmaking becomes big business, citizen cartographers are creating free personal alternatives. *Guardian, online newspaper*, (April 7, 2005). http://technology.guardian.co.uk/online/story/0,3605,1453293,00.html.

Duchêne, C. (2003). Automated map generalisation using communicating agents. In *Proceedings of 21st International Cartographic Conference*, pages 160–169, Durban, South Africa.

Dunkars, M. (2004). *Multiple representation databases for topographic information.* PhD thesis, KTH, Infrastructure, Stockholm.

Dutton, G. (1999). Scale, sinuosity and point selection in digital line generalization. *Cartography and Geographic Information Systems*, 26(1):33–53.

Dutton, G., Weibel, R., Peter, B., Bader, M., and Brazile, F. (1998). Agent workpackage a2 - constraint analysis. Technical report, AGENT consortium.

Dykes, J., MacEachren, A. M., and Kraak, M.-J., editors (2005). *Exploring geovisualization*. Elsevier, Amsterdam.

Edwardes, A., Burghardt, D., Bobzien, M., Harrie, L., Lehto, L., Reichenbacher, T., Sester, M., and Weibel, R. (2003). Map generalisation technology: Addressing the need for a common research platform. In *Proceedings of the 21st International Cartographic Conference*, Durban, South Africa.

Edwardes, A., Burghardt, D., and Neun, M. (2005). Interoperability in map generalisation research. In *Proceedings International Symposium on Generalization of Information (ISGI)*, Berlin.

Edwardes, A., Burghardt, D., and Neun, M. (2007). Experiments to build an open generalisation system. In Mackaness, W., Ruas, A., and Sarjakoski, T., editors, *Generalisation of geographic Information: Cartographic Modelling and Applications*, chapter 8, pages 161–175. Elsevier Science, Amsterdam.

Edwardes, A. and Mackaness, W. (1999). Modelling knowledge for automated generalization of categorical maps - a constraint based approach. In Atkinson, P. and Martin, D., editors, *Innovations in GIS 7, GeoComputation*, chapter 12, pages 161–173. Taylor&Francis, London.

Egenhofer, M.-J., Clementini, E., and Felice, P.-D. (1994). Evaluating inconsistencies among multiple representations. In *Sixth International Symposium on Spatial Data Handling*, volume 2, pages 901–920, Edinburgh, Scotland.

Ehrliholzer, R. (1995). Quality assessment in generalization: Integrating quantitative and qualitative methods. In *Proceedings of the 17th International Cartographic Conference*, pages 2241–2250, Barcelona.

Englander, R. (2002). *Java and SOAP*. O'Reilly, Sebastopol, CA.

ESRI (2007). ArcGIS Server – comprehensive server based gis. http://www.esri.com/arcgisserver (accessed 04/2007).

Feiler, J. (2000). *Application server: Powering the Web-Based Enterprise*. Academic Press, San Diego.

Fitzke, J., Greve, K., M., M., and Poth, A. (2004). Building sdis with free software – the deegree project. In *Proceedings 7th Conf. Global Spatial Data Infrastructure*, Bangalore, India.

Foerster, T. and Stoter, J. (2006). Establishing an ogc web processing service for generalization processes. In *9th ICA workshop on Generalisation and Multiple Representation*, Portland.

Frank, A. U. and Timpf, S. (1994). Multiple representations for cartographic objects in a multi-scale tree - an intelligent graphical zoom. *Computers and Graphics*, 18(6):348–376.

Gahegan, M. (2005). Beyond tools: visual support for the entire process of giscience. In MacEachren, A. and Kraak, M. J., editors, *Exploring GeoVisualization*, pages 83–99. Elsevier, Amsterdam.

Galanda, M. (2003). *Automated Polygon Generalization in a Multi Agent System*. PhD thesis, Department of Geography, University of Zurich.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns, Elements of Reusable Object-Oriented Software*. Addison Wesley Publishing Company.

Ghezzi, C., Jazayeri, M., and Mandrioli, D. (2003). *Fundamentals of software engineering*. Prentice Hall, Upper Saddle River, N.J.

Glass, G. (2002). *The Evolution of Web Services*. Prentice Hall.

Gold, C. M. (1999). Crust and anti-crust: A one-step boundary and skeleton extraction algorithm. In *ACM Symposium on Computational Geometry*, Miami.

Goodchild, M. (2005). Gis and modeling overview. In Maguire, D., Batty, M., and Goodchild, M., editors, *GIS, Spatial Analysis, and Modeling*, page 1–18. ESRI Press, Redlands, CA.

Goodchild, M., Egenhofer, M., Fegeas, R., and Kottman, C. (1999). *Interoperating Geographic Information Systems*. Kluwer Academic Publishers, Boston.

Goodchild, M. F. (1992). Geographical information science. *International Journal of Geographical Information Science*, 6(1):31–45.

Goodwin, J. (2005). What have ontologies ever done for us - potential applications at a national mapping agency. In *OWL: Experiences and Directions*, Galway, Ireland. http://www.mindswap.org/OWLWorkshop/sub18.pdf.

Gruber, T. R. (1993). A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199–220.

Hampe, M., Sester, M., and Harrie, L. (2004). Multiple representation databases to support visualization on mobile devices. In *Proceedings of the XXth ISPRS Congress*, International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences. XXXV(B4:IV), pages 135–140, Istanbul, Turkey.

Hangouët, J. (1998). *Approche et méthodes poul l'automatisation de la généralisation cartographique; application en bord de ville*. PhD thesis, Marne la Valée.

Hardy, P., Hayles, M., and Revell, P. (2003). Clarity - a new environment for generalisation using agents, java, xml and topology. In *Proceedings 5th Workshop on Progress in Automated Map Generalization*, Paris, France.

Harrie, L. (2001). *An Optimisation Approach to Cartographic Generalization*. PhD thesis, Lund Institute of Technology.

Harrie, L. (2005). Graphic generalisation methods in a system for real-time maps. In Hauska, H. and (eds.), H. T., editors, *ScanGIS'2005 - Proceedings of the 10th Scandinavian Research Conference on Geographical Information Sciences*.

Harrie, L. and Johansson, M. (2003). Real-time data generalisation and integration using java. *Geoforum Perspektiv*, pages 29–34.

Harrie, L. and Weibel, R. (2007). Modelling the overall process of generalisation. In Mackaness, W., Ruas, A., and Sarjakoski, T., editors, *Generalisation of geographic Information: Cartographic Modelling and Applications*, chapter 4, pages 67–87. Elsevier Science, Amsterdam.

Harrower, M. and Bloch, M. (2006). MapShaper.org: A map generalization web service. *IEEE Computer Graphics and Applications*, 26(4):22–27.

Haunert, J.-H. and Sester, M. (2004). Using the straight skeleton for generalisation in a multiple representation environment. In *7th ICA Workshop on Generalisation and Multiple Representation*, Leicester. http://ica.ign.fr (accessed 09/2006).

Hearnshaw, H. and Unwin, D. (1994). *Visualization in Geographical Information Systems*. John Wiley & Sons, New York.

Hojholt, P. (2000). Solving space conflicts in map generalization: Using a finite element method. *Cartography and Geographic Information Science*, 27(1):65–74.

ICA (1973). *Multilingual Dictionary of Technical Terms in Cartography*. Franz Steiner Verlag, Wiesbaden.

ICA (2004). Brain storming sessions at the ICA workshop on generalisation and multiple representation. available from http://ica.ign.fr/.

Illert, A. and Afflerbach, S. (2004). Global schema specification, GiMoDig-project, deliverable d5.3.1. Public EC report, http://gimodig.fgi.fi/deliverables.php (accessed 05/2007).

Jones, C., Purves, R., Ruas, A., Sanderson, M., Sester, M., van Kreveld, M., and Weibel, R. (2002). Spatial information retrieval and geographical ontologies: An overview of the spirit project. In *SIGIR 2002: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 387–388, Tampere, Finland. ACM Press.

Jones, C. B., Bundy, G. L., and Ware, J. M. (1995). Map generalization with a triangulated data structure. *Cartography and Geographic Information Systems*, 22(4):317–331.

Jones, C. B., Kidner, D. B., Luo, L. Q., Bundy, G. L., and Ware, J. M. (1996). Database design for a multi-scale spatial information system. *International Journal of Geographic Information Systems*, 10(8):901–920.

JTS (2006). The JTS Topology Suite. http://www.vividsolutions.com/jts/jtshome.htm (accessed 04/2007).

JUMP (2006). The JUMP Unified Mapping Platform. http://www.jump-project.org (accessed 04/2007).

Kilpeläinen, T. (1997). *Multiple Representation and Generalization of Geo-Databases for Topographic Maps*. PhD thesis, Finnish Geodetic Institute, Helsinki University of Technology.

Kottmann, C. (1999). Semantics and information communities. The OpenGIS Abstract Specification, Open GIS Consortium (OGC). http://portal.opengeospatial.org/files/?artifact_id=902 (accessed 05/2007).

Kreiter, N. (2002). Multirepräsentationsdatenbank als Basis von topografischen Landeskarten. Master's thesis, Institute of Cartography, ETH Zürich.

Kuhn, W. (2005). Geospatial semantics: Why, of what, and how? *Journal on Data Semantics*, 3:1–24.

Lake, R., Burggraf, D., Trinic, M., and Rae, L. (2004). *GML Geography Mark-Up Language*. John Wiley & Sons, Southern Gate, Chichester.

Lehto, L. (2003). Final system architecture. Public deliverable, GiMoDig - geospatial info-mobility service by real-time data-integration and generalisation. http://gimodig.fgi.fi/deliverables.php (accessed 04/2007).

Lehto, L. and Kilpeläinen, T. (2000). Real-time generalisation of geodata in the web. *Archives of Photogrammetry and Remote Sensing*, XXXIII(Part B4):559–566.

Lehto, L. and Kilpeläinen, T. (2001). Generalizing xml-encoded spatial data on the web. In *Proceedings of 20th International Cartographic Conference*, volume 4, pages 2390–2396, Beijing.

Lehto, L. and Sarjakoski, T. (2004). An open service architecture for mobile cartographic applications. In *Location Based Services & TeleCartography. Proceedings of the Symposium 2004*, pages 141–145, Vienna University of Technology.

Lemmens, R. (2006). *Semantic interoperability of distributed geo-services*. PhD thesis, Netherlands Geodetic Commission. NCG Publications on Geodesy 63.

Lemmens, R., Wytzisk, A., de By, R., Granell, C., Gould, M., and van Oosterom, P. (2006). Integrating semantic and syntactic descriptions to chain geographic services. *IEEE Internet Computing*, 10(5):42–52.

Linden, G., Smith, B., and York, J. (2003). Amazon.com recommendations. item-to-item collaborative filtering. *IEEE Internet Computing*, 10(5):42–52.

Mackaness, W. (2006). Automated cartography in a bush of ghosts. *Cartography and Geographic Information Science*, 33(4):210–221.

Mackaness, W. A. and Beard, M. K. (1993). Use of graph theory to support map generalization. *Cartography and Geographic Information System, Vol. 20, No. 4*, pages 210–221.

Mannes, J. (2004). Dynamische kartographische visualisierung von location based services mittels svg. Master's thesis, Department of Geography, University of Zurich.

McMaster, R. (1986). A statistical analysis of mathematical measures for linear simplification. *The American Cartographer*, 13(2):103–116.

McMaster, R. and Shea, S. (1992). *Generalization in Digital Cartography*. Association of American Geographers, Washington, USA.

McMaster, R. B. and Usery, E. L., editors (2004). *A Research Agenda for Geographic Information Science*. CRC Press, Boca Raton.

Morgenstern, D. and Schürer, D. (1999). A concept for model generalization of digital land-scape models from finer to coarser resolution. In *Proceedings 19th International Cartographic Conference*, pages 1021–1028, Ottawa, Canada.

Müller, J. C. (1991). Generalization of spatial databases. In Maguire, D. J., Goodchild, M. F., and Rhind, D. W., editors, *Geographic Information Systems: Principles and Practice*, volume 1, pages 457–475. Longman, London.

Mustière, S. (1995). Mesures de la qualité de la généralisation du linéaire. Rapport de Stage de DESS, Université Paris I / Ecole Nationale Supérieure de Géographie.

Mustière, S. (2005). Cartographic generalization of roads in a local and adaptive approach: A knowledge acquisition problem. *International Journal of Geographic Information Systems*, 19(8):937–955.

Mustière, S. and Moulin, B. (2002). What is spatial context in cartographic generalisation? In *Proceedings Symposium on Geospatial Theory, Processing and Applications*, Ottawa, Canada. CD-ROM.

Mustière, S., Saitta, L., and Zucker, J.-D. (2000). Abstraction in cartographic generalization. In *Lecture Notes In Computer Science, Proceedings of the 12th International Symposium on Foundations of Intelligent Systems*, volume 1932. Springer-Verlag.

Neun, M. (2006). Comment on OGC request 28: Web Processing Service (WPS). Submitted as Public Comment to the OGC. . http://www.opengeospatial.org/ (accessed 4/2007).

Neun, M. and Burghardt, D. (2005). Web services for an open generalisation research platform. In *8th ICA Workshop on Generalisation and Multiple Representation*, A Coruña, Spain. http://ica.ign.fr/ (accessed 09/2006).

Neun, M., Burghardt, D., and Weibel, R. (2006). Spatial structures as generalisation support services. In *Joint ISPRS Workshop Workshop on Multiple Representation and Interoperability of Spatial Data*, Hannover, Germany.

Neun, M., Burghardt, D., and Weibel, R. (in press). Web service approaches for providing enriched data structures to generalisation operators. *International Journal of Geographic Information Systems*.

Neun, M., Burghardt, D., and Weibel, R. (submitted). Automated processing for map generalization with modular operator services. *Submitted to GeoInformatica*.

Neun, M. and Steiniger, S. (2005). Modelling cartographic relations for categorical maps. In *Proceedings of the XXII ICA Conference*, A Coruña, Spain.

Neun, M., Weibel, R., and Burghardt, D. (2004). Data enrichment for adaptive generalisation. In *7th ICA Workshop on Generalisation and Multiple Representation*, Leicester. http://ica.ign.fr (accessed 09/2006).

OASIS (2002). Universal Description, Discovery, and Integration UDDI. http://www.uddi.org/ (accessed 04/2007).

OGC (1996). *The OpenGIS© Guide: Introduction to Interoperable Geoprocessing*. Beuhler, K. and McKe, L. (eds.), Open GIS Consortium Inc., Waylard, Mass.

OGC (1999). The OpenGIS© Simple Features Implementation Specification for SQL. http://www.opengeospatial.org/specs/. (accessed 07/2006).

OGC (2002). The OpenGIS© Abstract Specification, Topic 12: OpenGIS Service Architecture, Version 4.3, OGC 02-112. http://www.opengeospatial.org/specs/ (accessed 07/2006).

OGC (2003). The OpenGIS© Reference model, version 0.1.2. abstract specification ogc 03-040. http://www.opengeospatial.org/specs/ (accessed 07/2006).

OGC (2004). The OpenGIS© Geography Markup Language (GML) Encoding Specification, Version 3.1.1, OGC 03-105r1. http://www.opengeospatial.org/specs/ (accessed 07/2006).

OGC (2005). The OpenGIS© Discussion Paper: Web Processing Service (WPS), Version 0.4, OGC 05-007r4. http://www.opengeospatial.org/specs/ (accessed 07/2006).

Peterson, M. P. (2006). *Maps and the Internet*. Elsevier, Amsterdam.

Petzold, I., Burghardt, D., and Bobzien, M. (2006). Workflow management and generalisation services. In *9th ICA workshop on Generalisation and Multiple Representation*, Portland.

Plazanet, C. (1996). *Enrichissement des bases de données géographiques: analyse de la géométrie des objets linéaires pour la généralisation cartographique (application au routes)*. PhD thesis, Université de Marne-la-Vallée.

Plazanet, C., Affholder, J.-G., and Fritsch, E. (1995). The importance of geometric modeling in linear feature generalization. *Cartography and Geographic Information Systems*, 22(4):291–305.

Rainsford, D. and Mackaness, W. (2002). Template matsching in support of generalisation of rural buildings. In Richardson, D. and van Oosterom, P., editors, *Advances in Spatial Data Handling. Proceedings 10th International Symposium on Spatial Data Handling*, pages 137–151. Springer, Berlin Heidelberg.

Regnauld, N. (2001). Contextual building typification in automated map generalization. *Algorithmica*, 30(2):312–333.

Regnauld, N. (2003). Algorithms for the amalgamation of topographic data. In *Proceedings of the 21st International Cartographic Conference*, Durban.

Regnauld, N. (2005). Spatial structures to support automatic generalisation. In *Proceedings XXII International Cartographic Conference*, A Coruña, Spain.

Regnauld, N. (2006). Improving efficiency for developing automatic generalisation solutions. In *Joint ISPRS Workshop on Multiple Representation and Interoperability of Spatial Data*, Hannover.

Regnauld, N. (2007). Evolving from automating existing map production systems to producing maps on demand automatically. In *10th ICA/ACI Workshop on Generalisation and Multiple Representation*, Moskow, Russia.

Reichenbacher, T. (2004). *Mobile Cartography – Adaptive Visualisation of Geographic Information on Mobile Devices*. PhD thesis, Lehrstuhl für Kartographie der Universität München. Verlag Dr. Hut, München.

Richardson, D. and Müller, J. (1991). Towards rule-based selection of spatial objects for small scale map generalization. In B, B. and R, M., editors, *Map Generalization: Making Rules for Knowledge Representation. London*, page 136–149. Longman, London.

Riedemann, C. and Timm, C. (2003). Services for data integration. *Data Science Journal*, 2:75–83.

Ruas, A. (1995). Multiple paradigms for automated map generalization: Geometry, topology, hierarchical partitioning and local triangulation. In *Proceedings AutoCarto 12*, pages 69–78, Charlotte, USA.

Ruas, A. (1998). A method for buiding displacement in automated map generalisation. *International Journal of Geographical Information Science*, 12(8):789–803.

Ruas, A. and Lagrange, J. P. (1995). Data and knowledge modelling for generalization. In Müller, J. C., Lagrange, J. P., and Weibel, R., editors, *GIS and Generalization. Methodology and Practice. GISDATA 1*, pages 73–90. Taylor&Francis.

Ruas, A. and Plazanet, C. (1996). Strategies for automated generalization. In *Proceedings 7th International Symposium on Spatial Data Handling (=Advances in GIS Research II)*, pages 6.1–6.17, Delft. Taylor&Francis.

Sarjakoski, T., Sester, M., Sarjakoski, L., Harrie, L., Hampe, M., Lehto, L., and Koivula, T. (2005). Web generalisation services in GiMoDig - towards a standardised service for real-time generalisation. In Toppen, F. and Painho, M., editors, *8th AGILE Conference on GIScience*, Estoril, Portugal.

Schut, P. and Keens, S., editors (2005). *Web Processing Service (WPS) Interoperability Experiment: Final Report*. OGC 05-051. http://www.opengeospatial.org/specs/ (accessed 05/2007).

Sester, M. (2000). Generalization based on least squares adjustment. In *International Archives of Photogrammetry and Remote Sensing*, volume 33. ISPRS.

Sester, M., Sarjakoski, L. T., Harrie, L., Hampe, M., Koivula, T., Sarjakoski, T., Lehto, L., Elias, B., Nivala, A.-M., and Stigmar, H. (2004). Real-time generalisation and multiple representation in the gimodig mobile service, gimodig-project, deliverables d7.1.1, d7.2.1 and d7.3.1. Public EC report, http://gimodig.fgi.fi/deliverables.php. (accessed 07/2006).

Solutions, V. (2007). Jts topology suite and jcs conflation suite. http://www.vividsolutions.com/ (accessed 05/2007).

Spaccapietra, S., Parent, C., and Vangenot, C. (2000). GIS Databases: From Multiscale to MultiRepresentation. In B.Y.Choueiry and (Eds.), T., editors, *Abstraction, Reformulation, and Approximation*. Springer.

Steiniger, S. (2007). *Enabling Pattern Aware Map Generalization*. PhD thesis, Department of Geography, University of Zurich.

Steiniger, S., Lange, T., Burghardt, d., and Weibel, r. (in press). An approach for the classification of urban building structures based on discriminant analysis techniques. *Transactions in GIS*.

Steiniger, S. and Weibel, R. (2007). Relations between map objects in cartographic generalization. *Cartography and Geographic Information Science*. to appear in 2007.

Timpf, S. (1998). *Hierarchical Structures in Map Series*. PhD thesis, Technical University of Vienna, Austria.

Timpf, S. (2001). Geographic task models for geographic information processing. In Duckham, M. and Worboys, M., editors, *Meeting on Fundamental Questions in Geographic Information Science*, pages 217–229, Manchester, UK. http://www.geo.unizh.ch/~timpf/docs/Timpf_TaskOntologies.pdf (accessed 05/2007).

Tsang, E. (1993). *Foundations of Constraint Satisfaction*. Academic Press, London.

Tuvesson, H. and Harrie, L. (2003). Integration of navigational and cartographic data. In *Proceedings of 21st International Cartographic Conference*, Durban, South Africa.

van Oosterom, P. (2005). Variable-scale topological data structures suitable for progressive data transfer: The gapface tree and gap-edge forest. *Cartography and Geographic Information Science*, 32(4):331–346.

van Oosterom, P. and Laffra, C. (1990). Persistent graphical objects in procol. In *Proceedings TOOLS'90*, pages 271–283, Paris, France.

van Oosterom, P. and Schenkelaars, V. (1995). The development of a multi-scale gis. *International Journal of Geographic Information Systems*, 9(5):489–508.

Vckovski, A. (1998). *Interoperable and distributed processing in GIS*. Taylor and Francis, London.

Vijlbrief, T. and van Oosterom, P. (1992). Geo++: An extensible gis. In *Proceedings 5th International Symposium on Spatial Data Handling*, pages 40–50, Charleston, South Carolina.

W3C (1999). Hypertext Transfer Protocol, HTTP/1.1 specification. http://www.w3.org/Protocols/rfc2616/rfc2616.html (accessed 05/2007).

W3C (2004). Web Services Architecture, W3C working group note 11 february 2004. http://www.w3.org/TR/ws-arch/ (accessed 09/2006).

W3C (2006). Extensible Markup Language (XML), W3C recommendation. http://www.w3.org/TR/xml/ (accessed 05/2007).

Ware, J., Jones, C., and Thomas, N. (2003). Automated map generalization with multiple operators: A simulated annealing approach. *International Journal of Geographical Information Science*, 17(8):743–769.

Ware, J. M. and Jones, C. B. (1998). Conflict reduction in map generalization using iterative improvement. *GeoInformatica*, 2(4):383–407.

Weibel, R. (1991). Amplified intelligence and rule-based systems. In Buttenfield, B. P. and McMaster, R. B., editors, *Map Generalization: Making Rules for Knowledge Representation*, pages 172–186. Longman, London.

Weibel, R. (1997a). Generalization of spatial data: Principles and selected algorithms. In van Kreveld, M., Nievergelt, J., Roos, T., and Widmayer, P., editors, *Algorithmic Foundations of Geographic Information Systems*, pages 99–152. Springer.

Weibel, R. (1997b). A typology of constraints to line simplification. In Kraak, M. J. and Molenaar, M., editors, *Advances in GIS Research II*. Taylor and Francis.

Weibel, R. and Burghardt, D. (in press). On-the-fly generalization. In Kemp, K., editor, *Encyclopedia of Geographic Information Science*. Sage Publications.

Weibel, R. and Burhardt, D. (2003). Data enrichment for adaptive generalization. Proposal for project DEGEN, Departement of Geography, University of Zurich.

Weibel, R. and Dutton, G. (1998). Constraint-based automated map generalization. In *Proceedings 8th International Symposium on Spatial Data Handling*, pages 214–224.

Weibel, R. and Dutton, G. (1999). Generalising spatial data and dealing with multiple representations. In Longley, P. A., Goodchild, M. F., Maguire, D. J., and Rhind, D. W., editors, *Geographical Information Systems*, volume 1, pages 125–155. John Wiley & Sons, Inc.

Weibel, R., Keller, S., and Reichenbacher, T. (1995). Overcoming the knowledge acquisition bottleneck in map generalization: the role of of interactive systems and computational intelligence. In *Proceedings COSIT'95*, pages 139–156, Semmering, Austria.

Wilson, I. D., Ware, J. M., , and Ware, J. A. (2003). A genetic algorithm approach to cartographic map generalisation. *Computers in Industry*, 52(3):291–304.

Winer, D. (1999). XML-RPC specification. http://www.xmlrpc.com/stories/storyReader$7 (accessed 04/2007).

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

Zoraster, S. (1986). Integer programming applied to the map label placement problem. *Cartographica*.

Part II

Research Papers

Research Paper 1

Edwardes, A., D. Burghardt, M. Neun (2007). Experiments to build an open generalisation system. In: W. Mackaness, A. Ruas and T. Sarjakoski (eds), *Generalisation of geographic Information: Cartographic Modelling and Applications* chapter 8, (pp. 161-175). Amsterdam: Elsevier Science.

8 EXPERIMENTS IN BUILDING AN OPEN GENERALISATION SYSTEM

Alistair Edwardes, Dirk Burghardt and Moritz Neun
{aje | burg | neun} @ geo.unizh.ch

Department of Geography,
University of Zurich,
Winterthurerstr. 190 CH-8057
Zurich, Switzerland

Abstract
The increasing complexity of methods used in generalisation research together with growing demands for generalisation processing to support myriad new geospatial services and technologies has led researchers to investigate how open architectures might better support such changing needs. The chapter is motivated by two such requirements: to share generalisation techniques amongst researchers within the field and to present generalisation functionality externally to geographic services. The chapter first discusses the issue of openness in relation to the requirements of the generalisation community and then in the broader context of open standards, open architectures and open source for geographic information. Two open architectures for providing access to generalisation processing are then discussed based on these considerations. The first takes the approach of a middleware component that presents generalisation functionality through a coarse, web mapping interface. The second adopts the model of a web-service registry that provides access to generalisation operations in a distributed and platform independent way. Implementations of the two systems are then described. The middleware approach reuses existing protocols for requesting maps and representing map specifications and geographic phenomena. It is based on open-source technologies for its core infrastructure. The registry model implements a set of web services standards that allow generalisation researchers to register and expose their operations. These can then be accessed independent of language bindings or of the location where the code is being executed. Interaction with the registry is possible with plug-ins implemented for different desktop mapping software or via dynamically generated web information. In its conclusion the chapter considers the successes of each approach and the main limitations. Foremost of these is the lack of a formal conceptualisation for the generalisation domain.

Keywords

open architectures , open systems, standards, protocols, open source, on-demand mapping, on-the-fly generalisation, web mapping, location-based services, web services, publish-find-bind, Open Geospatial Consortium, middleware, registry, common research platform, portrayal model, point-of-interest maps, interoperability

8.1 Introduction

There is a growing consensus within the map generalisation research community that open research platforms will allow closer integration in terms of collaborative research, data abstractions, interoperability of functional components and the augmentation of geo-spatial applications with generalisation capabilities. This desire has been evidenced through discussions at the various meetings of the ICA Commission on Generalisation and Multiple Representation (Beijing 2001, Ottawa 2002, Paris 2003, Leicester 2004), described in Edwardes *et al.,* (2003).

This chapter describes the authors' experiences in developing systems to support collaborative research in map generalisation. Two models using open architectures to facilitate cooperation amongst researchers are described. In the first openness is explored in relation to the interfaces and concepts of mapping services. In the second, opening access to generalisation techniques using web services is investigated.

The chapter provides insights into models for designing open systems and the concepts used by these. It also describes how standards for geographic information concepts and services can be used to help share generalisation research. It then presents practical experiences gained from implementing such systems and suggests useful technologies to assist the development of these. Finally, it discusses interoperability issues associated with improving collaboration amongst generalisation researchers and makes suggestions as to the best way for future progress.

8.1.1. Motivation

The growth of new technologies for the retrieval, handling and visualisation of geo-spatial data has brought with it increasing demands on automated cartography to provide theory and methods that will allow their coherent operation (Meng, 2003). As well as classical needs for cartographic generalisation, new challenges are raised by the necessity for more usable (Nivala et al., 2003), flexible portrayals of geographic data (Barkowsky et al., 2000; Avelar and Müller, 2000; Elias 2002), matching different levels of interaction (Crampton, 2002), device capabilities (Chalmers et al., 2001; Arleth, 1999), modes of use (Fuhrmann and Kuhn, 1998) and needs of users for geographic information (Raper *et al.* 2002). These include: The provision of cartographic products on devices with differing display capabilities such as personal computers and small screen devices (Sarjakoski and Sarjakoski, this volume (Chapter 8); the 'ego-centric' adaptation of information portrayal according to an individual's context and preferences (Zipf and Richter, 2002; Reichenbacher, 2003; 2004) and activities (Sester and Elias, this volume (Chapter 10); dynamic geo-spatial information retrieval and data conflation for 'Point of Interest' mapping (Arikawa et al., 1994; Edwardes and Burghardt, 2004).

At the same time, the techniques used in more conventional generalisation research have become increasingly more complex. Algorithm development has become more focussed on satisfying constraints and modelling knowledge (Beard 1991; Weibel 1997; Weibel and Dutton 1998; Ruas 1999). This requires both the intelligence to optimise amongst numerous design constraints (Burghardt and Meier 1997; Ware and Jones 1998; Sester 2000; Harrie and Sarjakoski, 2002) and better representations of geographic phenomena, space and spatial relations (Ruas and Plazanet, 1996,

Regnauld 1996; Mustière 2001). Considerable effort is needed to meet these requirements. Researchers must spend significant amounts of valuable time gathering together tools, designing a platform, integrating tools etc. just to reach the research frontier. Moreover, platforms developed through this process tend to be institute specific and differ greatly, leaving little opportunity for sharing research.

8.1.2. Requirements for an open generalisation system

These motivations highlight two areas where openness in research systems could enhance collaboration and cooperation in research. On the one hand, there is the need to support cooperation by sharing of techniques *within* the research community, for example new algorithms, data structures, measures and control architectures. On the other, there is the need to support external collaboration through the *application* of generalisation in other GIS research areas, for example in on-demand mapping, geo-visualisation and location-based services. Within the community openness needs to be at very detailed and technical levels. Outside openness should be to support the inclusion of generalisation concepts and techniques within the broader body of GI research.

Internal to generalisation research, requirements include:

1) To support sharing and comparing of techniques and results;
2) To allow researchers to share and access complex spatial data-modelling, analysis and re-structuring functionality so that these may be used as the basis for new techniques;
3) To enable access to libraries of algorithms that can be used to study the procedural knowledge required for sequencing and orchestrating generalisation operations;
4) To allow researchers to test and demonstrate new or improved functionality within a holistic generalisation framework made from shared common components.

Externally they include:

1) To allow generalisation functionality to be presented and accessed in such a way that it can be integrated transparently with geographic applications from other research areas;
2) To express generalisation concepts, such as map constraints, multi-scale representations of information, generalisation operations and alternative portrayal types, in a formal machine-readable manner;
3) To perform generalisation in real-time on transient data (e.g. search results or the result of a route calculation) respecting both dynamic restrictions, such as the map user's current location or the time of day, and static associations, such as inherent spatial relationships between the retrieved features and the base map features;
4) To link between sets of multi-scale representations of the same data.

In addition, the two areas also have many requirements in common, for example:

1) The ability to support researchers independent of their choice of platform or development environment;
2) The ability to access, encode and transfer data without loss of information;
3) The ability to describe and encode map specifications and styling rules.

8.2 Context and Evolution in Computer Science

8.2.1 Open architectures

Open Architectures are collections of components and their interfaces whose specification, at some level of granularity, has been made public. Interfaces can be thought of as contracts setting out the obligations between a component's user and provider (Vckovski, 1998). To share generalisation techniques amongst researchers interfaces need detailed definitions, presenting many generalisation operations and their parameters. For generalisation applications, interfaces may be coarser, perhaps with only a single "Generalise Map" operation. Open architectures based on lightweight distributed platforms have attracted significant interest in the generalisation community (Lehto and Kilpeläinen, 2000, 2001; Harrie and Johansson, 2003; Badard and Braun 2003; Burghardt et al., 2005).

8.2.2 Open Protocols

Open protocols provide the language syntax and concept formalisation for communicating within an open architecture. Protocols are defined to encode data modelling concepts relevant to the domain and the operations of an interface. They may define formats for describing an operation, for encoding information to be exchanged across the interface or for coordinating the communication. Protocols should be based on a single domain abstractions common to all relevant components, though their logical encoding (e.g. Java classes, XML, SQL structures) can differ throughout the architecture. For generalisation research, protocols must be able to represent models of geographic phenomena (Sondheim et al., 1999) that encapsulate geometric, topological and semantic concepts and which may be encoded as parameters of operations. To present a generalisation service to other applications protocols must also represent cartography concepts such as: map specifications (e.g. map extent and scale), map content and styling, map theme and spatial region-of-interest, and different map types.

8.2.3. Open standards

Open standards formalise the definitions of open architectures. They are defined through open, international, participatory processes, are publicly documented and available to use without licensing or royalty restrictions. Standardisation eases interoperability between components from different researchers (Hjelm, 2002) and allows platforms to be designed without reference to particular software. Most importantly, they formalise definitions for geo-spatial data abstractions, easing communication amongst researchers and minimising duplication in design decisions. In this regard, the work of standardisation bodies such as the Open Geospatial Consortium (OGC, 2004) and the World-Wide Web Consortium (W3C, 2004) has revolutionised how GI Science is now practised. There are many standards that relate to open generalisation systems. Table 8.1 illustrates some of the more salient ones.

	Standard	Description.
Mapping	Web Map Service (WMS 2004)	The WMS specification defines an interface to allow mapping services to be made accessible over the web.
	Styled Layer Descriptors (SLD 2004)	The SLD allows map specifications to be defined. It encodes concepts for specifying the content, the map 'layers', and presentation, the layer 'styles' of a map.
	Scalable Vector Graphics (SVG 2004)	SVG is an XML encoding to describe graphics using vector primitives. (c.f. Lehto and Kilpeläinen 2000; Cecconi and Galanda 2002; Reichenbacher 2002; Takagi *et al.,* 2003).
Data Handling	Web Feature Service (WFS 2004)	The WFS specification defines an interface for accessing spatial data, as geographic features, over the web.
	Geography Markup Language (GML 2004)	GML is a protocol for encoding geographic feature descriptions in XML. GML uses standard conceptual abstractions (ISO 19107; ISO 19109) as a data schema for classifying features, their attributes and geometries.
	Filter Encoding (FES 2004)	The filter encoding specification provides a neutral protocol for constraining spatial and semantic queries on spatial data resources (e.g. a WFS).
Web Services	Web Services Architecture (WSA 2004) - Simple Object Access Protocol (SOAP 2003) - Web Services Definition Language (WSDL 2001)	The Web Services Architecture (WSA) is a collection of protocols and standards to allow software applications to interoperate over the Internet in a platform independent manner. Two of these are: - SOAP is a light-weight implementation of a WSA protocol. It allows access to objects, operations or data over a network using structured messages in XML. - WSDL a WSA protocol to describe the interface of a Web Service. It allows the service to expose the operations and message formats it supports.

Table 8.1. Current relevant standardisation efforts.

The WMS and WFS specifications aid the deployment of generalisation functionality through geographic application services. The SLD and FES protocols allow requests to these services to be characterised and provide a mechanism for triggering generalisation. GML is a protocol for encoding and exchanging spatial data. It is important for both coarse and fine-grained systems. For mapping and data services it encodes query responses, describes transient data to be portrayed dynamically and describes regions of interest related to the map theme and user. For fine grained operations it describes parameter data types. The web services standards are generic mechanisms for accessing computational objects and operations over the web. In a platform for sharing research the main aim is to provide access techniques for comparison rather than to provide a framework of operations that can be integrated into a complete system. Hence, these standards can be used to express generalisation operations at fine-grained atomic levels. These can be accessed in a neutral XML format, independent of particular development environments.

8.2.4. Open source

There are many definitions of open source software (c.f. OSI, 2004), mainly differing in their description of licensing conditions and how the software can be modified and redistributed. Typically the software is provided together with its source code. Such software might therefore be considered as an open architecture that has been specified at a very detailed level. Because open source software can be modified, it is highly adaptable and can be closely integrated with other custom code. It is also often provided free of charge. Often research efforts can be hastened because open source tools can supply core functionality necessary to build a mapping platform, which is otherwise not related to research aims. A number of GIS open source projects exist

that are of interest to the field of generalisation. Some examples of these are listed in Table 8.2.

Software	Description.
GeoAPI (2004)	GeoAPI implements OGC protocols for geographic information in Java. This simplifies the low level integration amongst the different frameworks described below. The work is being made in concert with the OGC Geographic Objects initiative (GO-1 2004).
Deegree (2004), GeoTools (2004), GeoServer (2004)	These are different Java frameworks implementing OGC Web Services specifications (including WMS and WFS).
JTS Topology Suite (JTS), JTS Conflation Suite (JCS) and JUMP (Vivid Solutions, 2004)	JTS and JCS are tools for spatial analysis and spatial data structuring. They use standard based geometry abstractions. (Harrie and Johansson, 2003; Sarjakoski et. al, 2005). JUMP is a desktop mapping platform that uses JTS and JCS.

Table 8.2. Relevant open-source projects.

Amongst the most useful types of open source software are *frameworks*. Frameworks define architectures that modularise functionality through the definition of responsibilities and collaboration. Their aim is to encourage the reuse of the architectural design in order to create similar types of software rather than simply encourage the reuse of code (Gamma *et al.,* 1995). Frameworks are often implemented in response to open standards, with their modular design enabling compliance at very fine levels of granularity. GeoApi is an example of this (Table 8.2). The Frameworks implementing the WMS standards are also particularly useful for an open generalisation system since much of their design can be reused.

8.3 Architectural Models for Open Generalisation Systems.

The two types of research requirements suggest two different architectural approaches. Generalisation provided as part of a wider portrayal strategy is best deployed through extensions to existing models for geographic services. This is because these use interfaces and protocols that are standardized and well understood and open-source implementations mean that developers can focus on augmenting existing systems rather than starting from scratch. This sort of deployment is termed *middleware*, since the services sit in the middle of the architecture, between client applications and data servers. Research requiring atomic access to generalisation techniques is better supported by services that can expose operations and their parameters in very detailed ways. The web services model (a set of operations that can be accessed over the Internet using XML, possibly via an intermediary *registry*), provides a good method to meet these needs. This is because: 1) operations, offered by the generalization services, can be described and used independently of language bindings, 2) data types for parameters can be described using existing standards for geographic information exchange, and 3) functionality can be shared without users needing physical access to either the code implementing the operation or server performing the computation.

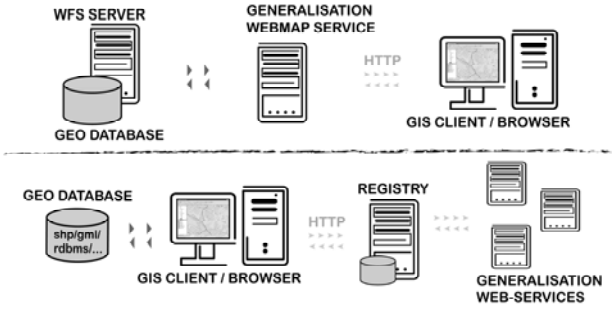


Figure 8.1. Two types of Generalisation Service: Middleware (top half) and Registry (lower half).

Figure 8.1 illustrates the two types of generalisation services. In the first scenario access to the service is by specifying a map, with the service handling its own data requirements (Lehto and Sarjakoski, 2004; Illert and Afflerbach, 2004). In the second case the client must handle its own data and data validation issues. Additionally, the mode of interaction in each scenario differs. The middleware model runs completely automatically whereas in the registry model the user has more interactive control.

8.3.1. Middleware

A categorisation of basic services for architectures portraying spatial information is described by the OGC's 'Portrayal Model' (OGC, 2003: p.23) (also known as the 'Cuthbert Model' (Cuthbert, 1998)). It describes a pipeline of four sequential processing steps:

- FILTER: Accessing geographic features from a database through spatial and semantic filters;
- DEG (Display Element Generator): Combining geometric and semantic feature information with styling rules to generate styled graphical vector primitives (e.g., postscript instructions, SVG elements or, Java graphic objects);
- RENDER: Drawing the display elements on an image canvas. In essence, projection, clipping, rasterisation and anti-aliasing of graphic vectors (Foley *et al.,* 1996, Ch. 3 and 19);
- DISPLAY: Making rendered image visible on the output device (Foley *et al.,* 1996, Ch. 4).

Adopting the concepts of this decomposition is useful because it allows generalisation to be discussed within the broader context of portrayal. Several open source implementations for web mapping also follow this model as a *de facto* standard. Generalisation functionality can be added to this model at various points. Figure 8.2 illustrates this, presenting the portrayal model on the left with possible modifications in boxes on the right.

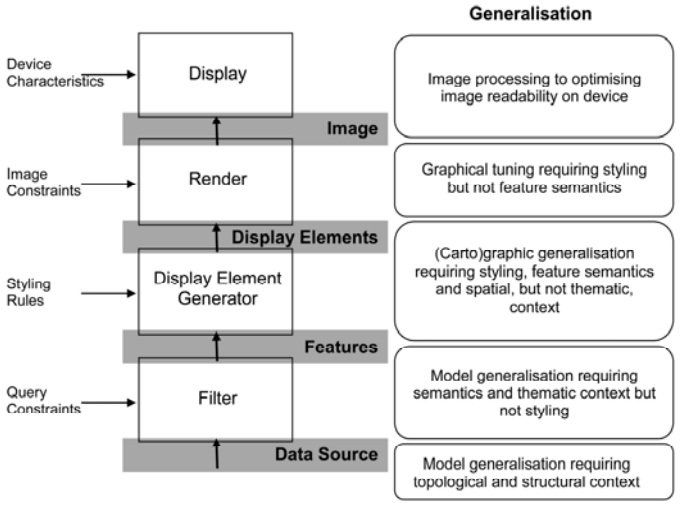


Figure 8.2. The Portrayal model and generalisation (adapted from Cuthbert (1998): Copyright Open Geospatial Consortium, Inc., and following the generalisation typology of Weibel and Dutton (1998)).

At the data source level the concern is with model generalisation with respect to data resolution and data volume rather than graphical generalisation, which would require associated styling information. Data enrichment can be undertaken to support this by modelling complex multi-scale spatial relationships (Neun *et al.,* 2004). Multiple representations of features and multi-resolution data can also be managed at this point (Weibel and Dutton 1999; Hampe *et al.,* 2003; Cecconi 2003). Filtering allows control over the map level-of-detail and theme. Layers modelling different phenomena can be instantiated by selection according to scale, semantics, spatial relationships and relative importance. The display element generation step provides the most suitable point for graphical generalisation. Information is available on both feature semantics and styling, allowing graphical legibility constraints to be effectively analysed. At the rendering stage, only styled graphical primitives without semantics exist. Techniques based on coordinate transformations can be applied, for example variable scale projections (Harrie *et al.,* 2002; Rappo *et al.,* 2004). Tuning to enhance the impression of graphical variables such as brightness and colour contrasts (Bertin, 1973) can also be performed. Text placement (Petzold *et al.,* 2003) might also be applied here, particularly if the language of text is localised dynamically on a client. At the display stage, there is no knowledge about the image content so generalisation is not performed. To some extent, legibility related to the client's display might be improved, for example by applying anti-aliasing using super-sampled images or automatic adaptation of display brightness and contrast according to background light conditions.

8.3.2. 'Publish-Find-Bind' – Registry for a research platform.

In an open research model the idea is that every researcher can deploy their own generalisation service. Through the Internet and the use of platform independent technologies such services can reside on servers all over the world. To discover these services a "Yellow Pages" is needed, indicating which services are available, where they are located and what algorithms they offer (Burghardt *et al.,* 2005). This is the

"Registry" model for generalisation services. The registry offers a single access-point where all further information can be found. Whilst services can change and move they can always be found again through the registry.

This model for sharing and discovering generalisation services can be summarized by the "publish-find-bind" paradigm (UDDI 2004), shown in Figure 8.3. The "publish" step is carried out by the service provider, e.g. a researcher who wants to make their generalisation operation available. They must create (a) an interface description containing the parameters of their generalisation service and the service endpoint, (an URL where the service can be accessed). Once the interface description is published (b) in the registry database, the community can access the service. The "find" step is done by a service consumer, e.g. a researcher who wants to test an operation. Having selected the desired service from those available (1), the link pointing to the interface description (2) is followed and the interface description itself retrieved (3). Using the interface description, the consumer can "bind" to a service and establish communication with the service-endpoint directly.

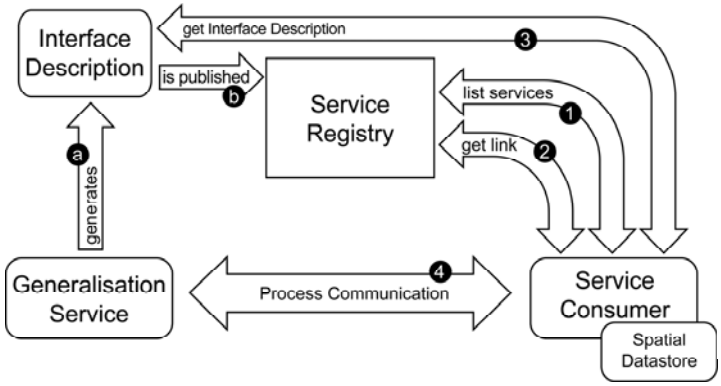


Figure 8.3. Registry for Generalisation Services.

Accessing the interface and operations of a generalisation service can either be through a form-based webpage or a plug-in for mapping software. Only a webpage client has the potential to upload a file via HTTP. A plug-in in a GIS integrates seamlessly within a cartographic application. The user can access the service without needing to export and re-import data because features are encoded and decoded directly within the application. Features sent to the service are encoded in GML and embedded within a SOAP message. The use of these XML formats (GML, SOAP) makes the approach very open and flexible. However, the costs of data conversions and transporting of large amounts of data can become a bottleneck in real-time applications (Gbei *et al.,* 2003), though for a research platform this is a minor problem.

Another way of supplying data to a service is with the URL of the data source (e.g. WFS) directly. The service then accesses the data source itself, processes the data, and sends it to the user (Sester *et al.,* 2004). In this context, the service-consumer is a simple client controlling the process without uploading cartographic data. Similar concepts to this, e.g. using CORBA (Common Object Request Broker Architecture), have been previously implemented in mapping platforms (Hardy *et al.,* 2003).

8.4 Implementation of the Two Architectures

Systems were implemented to demonstrate how openness could support research and help identify problems that this might entail. The first example implemented a generalisation service that dynamically generated points-of-interest maps for a location based service, as part of the EC project WebPark (Edwardes *et al.,* 2003). The second implementation demonstrated how a web services architecture could ease the sharing of techniques and results amongst researchers.

8.4.1. On-the-fly generalization in the WebPark project

a) Strategy
The aim of the project WebPark (2004) was to develop a mobile information system for visitors to natural and recreational areas. A particular emphasis in this research was the presentation of wildlife and "points of interest" information on small screen devices, using different forms of portrayal, scales and symbolisation (Edwardes and Burghardt 2004). A two-pronged approach to generalisation was taken to achieve this. Analysis of user needs for information indicated that some data was relatively static and some very dynamic. This meant data could be split into background (or base map) and foreground (or thematic) types. Background data helped to orientate users within the information context (e.g. topographic maps and maps of general animal distributions or ecology). Foreground information delivered dynamic content in response to user queries (e.g. points of interest such as restaurants and recent animal sightings).

b) Implementation
A generalisation service was constructed using Deegree software (Fitzke *et al.,* 2004). This was found to be a highly modular and adaptable framework. The generalisation of background information was performed offline using semi-automated approaches. This was organised in a database using multi-resolution data structures, and configured for access through a WMS using static, scale-dependent, layer definitions. Handling of dynamic data was enabled using mechanisms in the SLD. This allowed data definitions and data sources to be passed to the service at runtime. Data definitions could either be encoded using as FES (Filter Encoding Implementation Specification) constraints related to a WFS or data could be encoded in GML and passed in directly as a layer definition

The style definition in the SLD was used to state when and how a layer should be generalised. This was indicated through the semantics of the style element which could be configured using the `SemanticTypeIdentifier`. Customisation of the Deegree framework was required so these identifiers could be interpreted by the system. The framework permitted the behaviour of the "GetMap" operation to be configured at runtime using a custom map request handling class. Thus modifications could be made separately from the core framework code. Atomic generalisation operations were added using framework base-classes called display element *Optimizers*. Several operations could be applied sequentially using *Optimizer Chains*. Optimizers were created to perform on-the-fly typification of point-sets (Burghardt *et al.,* 2004), simplification of point-sets and lines (Visvalingam and Whyatt, 1993), point displacement (Edwardes, 2004) and density surface generation from points.

Figure 8.4 illustrates the places in the framework where configuration and customisation was made to add generalisation functionality

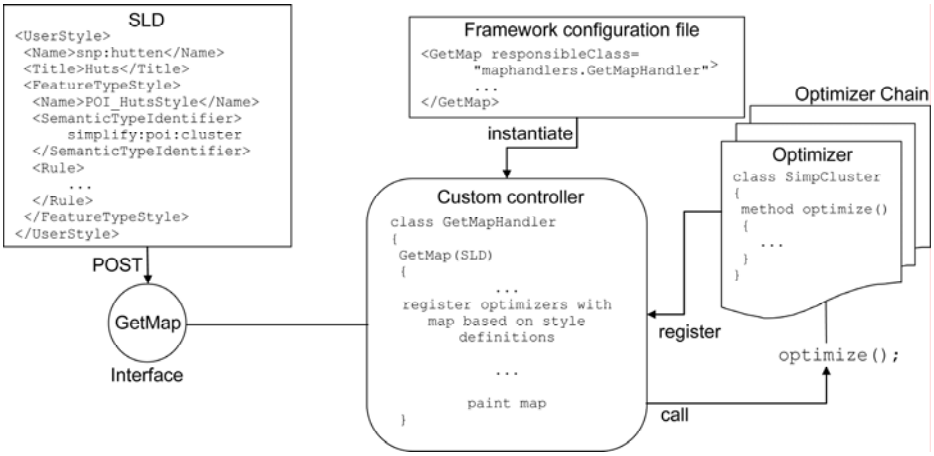
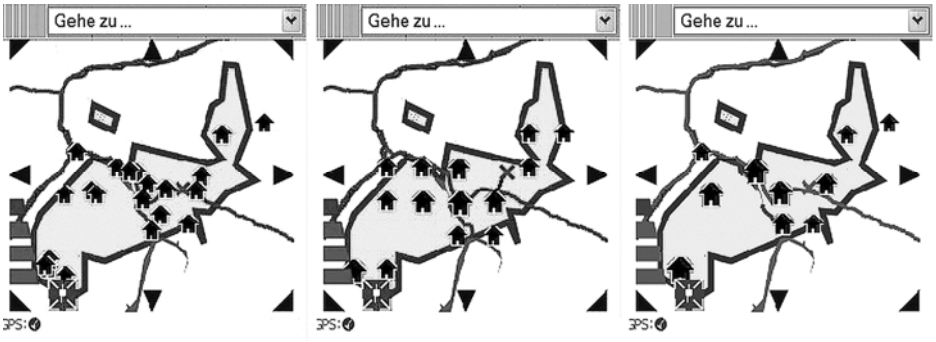


Figure 8.4. Configuration and customisation of Deegree to add generalisation functionality.

c) Results and Discussion

Figure 8.5 shows examples of point set generalisation. The first picture illustrates the overlapping of symbols (mountain huts in the Swiss National Park) without generalisation. The second, the same situation after applying an icon optimizer based on a quadtree approach (Burghardt *et al.,* 2004). The final picture shows the result of icon optimisation using a hierarchical clustering approach. The size of an icon is changed dependent on the number of underlying features that the icon represents.



(a) *(b)* *(c)*

Figure 8.5. Examples of graphical generalisation (a) original, b) quadtree typification, c) hierarchical cluster simplification).

The approach was straightforward to implement and relatively fast (around 100 milliseconds per operation). Layers provided meaningful groupings for features and free access to geometric, and semantic and styling information of display elements was possible. This provided a two level view of the features (grouped by layer and individually) which was useful for simple generalisation but less adequate when considering thematic relationships such as semantic hierarchies amongst layers, and spatial and topological relationships amongst features in different layers. The styling definitions and optimizers gave flexibility in map portrayal, though the approach of

chaining optimizers had limitations for complex generalisation. The sequence of processing (for example, first selection then displacement), had to be predefined, which limited how procedural intelligence could be incorporated in the system.

8.4.2 Implementation of an interactive generalisation platform

a) Strategy

The development was made as part of ongoing efforts by the ICA Commission on Map Generalisation and Multiple Representation to improve testing and sharing of generalisation algorithms (Badard and Braun 2003; Edwardes *et al.,* 2003).The goal of the platform was to help researchers share their techniques through the ICA web site (ICA, 2005), at the level of basic operators and measures, in a way that was independent of coding language and didn't require intellectual property to be exposed.

b) Implementation

The two usage scenarios, by web page and by mapping software plug-in (see Section 8.3.2), were implemented, using Java Servlets. The JUMP provided tools for working with Shapefiles in the browser example. For the handling of Shape and GML geometries JTS tools were used. The plug-in was developed for use with JUMP as the client. Other plug-ins, for platforms such as ArcView®, are planned as future project work. The client was implemented as an easy-to-use, interactive graphical user interface. The web page example uses standard HTML pages accessible by any standard browser. The user accesses the service through a start page containing all available services. This page is dynamically created with information from the service registry. After selecting a particular Generalisation Service the user was presented with a new, dynamically created page which allowed parameters for the algorithm to be entered and a Shapefile containing source data to be uploaded from the local system.

The JUMP plug-in has the same functionality as the browser solution but integrates seamlessly into the software, so that the user does not have to quit the application and does not notice a significant difference between using a local or remote algorithm. The plug-in integrates into the JUMP menu bar. It automatically checks every time it is started for all available generalisation services. The result of this search is displayed in a selection list to the user. Figure 8.6 illustrates this idea in the plug-in.

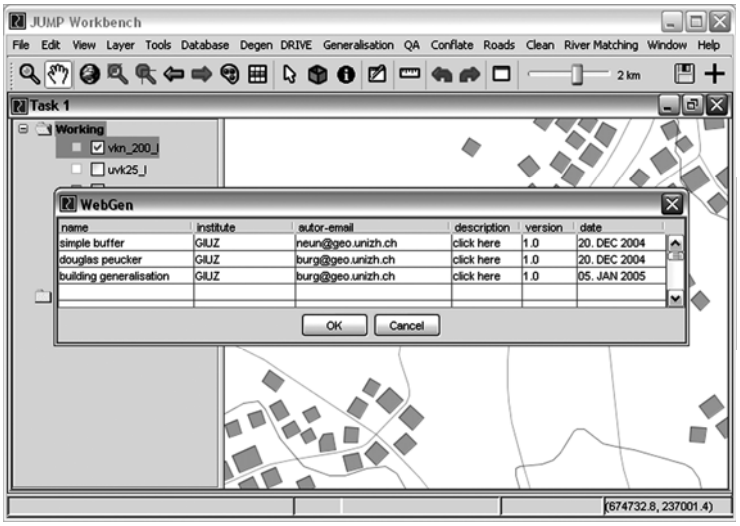


Figure 8.6. List of available services (from the Generalisation Service registry)

The user selects the desired operation from the list. They are then presented with an entry form for the corresponding algorithm's parameters. These entry forms are dynamically created using the interface description. An example of a simple interface description for a building simplification algorithm is shown in Figure 8.7. The data format for the interface description is XML, accessed via the Registry. This XML format extends WSDL with schema definitions for the simple features which are required by every generalisation operation. The registry uses the "publish-find-bind" concept of UDDI but is implemented as a more lightweight database front-end which can be queried for available services and then returns the XML descriptions.

```
<?xml version="1.0" encoding="UTF-8" ?>
<webgen xmlns:gml="http://www.opengis.net/gml" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/">
  <name>building simplification</name>
  <method>buildingSimplification</method>
  <endpoint>http://server/soap-endpoint</endpoint>
  <description>simplifies all buildings in a layer and returns resulting buildings</description>
  <config>
    <layer>
      <schema>
        <attribute name="geom" type="GEOMETRY">
          <allowed>gml:Polygon</allowed>
          <allowed>gml:MultiPolygon</allowed>
        </attribute>
      </schema>
    </layer>
    <param name="min edge length" type="SOAP-ENC:double">
      <description>Minimum Edge Length</description>
    </param>
  </config>
</webgen>
```

Figure 8.7. XML Interface description

After all the parameter information has been entered and the operation started, the plug-in establishes communication with the server. The communication process is summarised in Figure 8.8. It consists of passing SOAP requests and responses. The client sends the name of the desired generalisation operation, the operation parameters and the data to generalise to the server. The generalised data or an exception report is sent back in response.

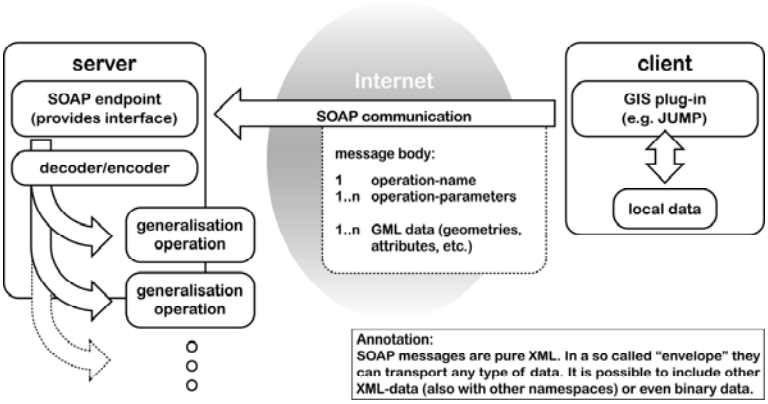


Figure 8.8. SOAP client-server communication

c) Results and Discussion

The implementation of client and server components effectively demonstrated an open generalisation service with a registry. The use of standard protocols, such as GML, and SLD, to define parameter data types meant that geographic phenomena and map specification (e.g. scale and styling) could be reasonably represented. In particular, they gave good assistance for sharing research techniques for geometric generalisation of independent features or amongst features of the same feature-type. Additional input and practical experience with the system by the research community is now needed to understand how the service can be improved. Open issues are; 1) how protocols can better express more contextual inter-object and inter-class spatial relationships, necessary for complex generalisation, 2) what generic methods can be found to represent semantic relationships between features, e.g. importance ordering and attribute similarity, and 3) how differences in terminology for feature-types and generalisation operation descriptions can be best resolved. There are various other possibilities for extension and improvement. In the context of managing an entire generalisation workflow or accessing a completely server-based generalisation system, solutions for extended session-management and data persistence on a server are being considered. This stateful behaviour could also be a way for flexibly accessing more complex agent based systems as well overcoming data transfer bottlenecks.

8.5 Conclusion

The research demonstrated two approaches to addressing the issue of improved access to research. In each case the use of standards and frameworks from the existing body of knowledge on architectural design was found to be highly desirable. This was partly for practical reasons, as it helped to focus research efforts by supplying auxiliary tools, but also because they provided technology neutral, standardised design concepts and protocols that made it easier to express where and how functionality could be deployed. However, it was clear that the standards also lacked concepts and methods required to undertake and share very complex generalisation research, for example modelling highly contextual geometric relationships or constructing sophisticated control structures for managing sequencing and orchestration of operations. For instance, simple constraints were handled implicitly by parameterising algorithms accordingly. Ideally, these would be specified independently in a more formal way (c.f. Edwardes and Mackaness 1999; Hardy *et*

al., 2003; Sarjakoski *et al.,* 2005). There is also wider semantic issue that needs to be addressed. Generalisation research often needs ontological descriptions about geographic phenomena. For example, the algorithm of Rainsford and Mackaness (2002) for generalising rural buildings needs to make explicit definitions about what constitutes a *rural building* to their algorithm. They 'define' this geometrically using template matching. Without such descriptions, assumptions must be made which lead to difficulties in reusing techniques if the two parties (designer and re-user) do not share a common understanding over the definition of feature type. These interoperability issues can only be resolved through consensus formalisation within the generalisation research information community (OGC 1999) and by practical experience in sharing research. In this regard the registry platform has strong contributions to make. Its relative simplicity, language neutrality, and ease for publishing and accessing algorithms means the barriers to researchers interoperating are very low. Thus such experiences can be shared fairly rapidly. In addition, because the owners of each algorithm remain in complete control of its code and the responsibility for its maintenance, intellectual property obstructions and administration issues are fewer and cooperation more feasible between commercial and non-commercial parties.

Acknowledgements

This work was in part carried out through the EC-IST Framework 5 project "WebPark: Geographically relevant information for mobile users in protected areas" (IST 2000-31041). The authors gratefully acknowledge the financial support of the Swiss Office of Education and Science (OFES) within the scope of this project (BBW Nr. 01.0187-1).

References

ARIKAWA, M., KAWAKITA, H., KAMBAYASHI, Y. 1994, Dynamic Maps as Composite Views of Varied Geographic Database Servers, In Proc. 1st Int. Conf. on Applications of Databases (Witold Litwin, Tore Rische Eds.), Lecture Notes in Computer Science 819, (Heidelberg, Berlin: Springer-Verlag), pp. 142- 157

ARLETH, M., 1999, Problems in screen map design. In Proc 19th (ICA/ACI) Int. Cartographic Conference, Ottawa Canada.

AVELAR, S. AND MÜLLER, M., 2000, Generating topologically correct schematic maps, In Proc. 9th Int. Symp. on Spatial Data Handling, pp. 4a.28–4a.35.

BADARD, T., AND BRAUN, A., 2003, OXYGENE: An open framework for the deployment of geographic web services, In Proc of the 21st Int. Cartographic Conf., Durban, South Africa.

BARKOWSKY, T., LATECKI, L. J., RICHTER K-F., 2000, Schematizing Maps: Simplification of Geographic Shape by Discrete Curve Evolution, Spatial Cognition II, Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications, LNAI 1849, (Berlin, Heidelberg, Berlin: Springer-Verlag), p.41-53,

BEARD, K.M., 1991, Constraints on rule formation. In Map Generalisation: Making rules for Knowledge Representation, Buttenfield B P, McMaster R B (eds), (London: Longman Scientific) pp. 121-135.

BERTIN, J., 1973, Sémiologie graphique, (Paris: Gauthier-Villars)

BURGHARDT, D. AND MEIER, S., 1997, Cartographic Displacement using the Snakes Concept. In, Semantic Modelling for the Acquisition of Topographic Information from Images and Maps, W. Foerstner and L. Pluemer (eds). (Basel: Birkhaeuser-Verlag)

BURGHARDT, D., NEUN, M. AND WEIBEL, R., 2005, Generalisation Services on the Web – A Classification and an Initial Prototype Implementation, In Proc. Auto-Carto 2005 Research Symposium, Las Vegas.

BURGHARDT, D., PURVES, R., EDWARDES, A., 2004, Techniques for on-the-fly generalisation of thematic point data using hierarchical data structures, In Proc of Geographical Information Systems Research – UK (GISRUK 2004), Norwich UK.
CECCONI, A., 2003, Integration of Cartographic Generalisation and Multi-Scale Databases for Enhanced Web Mapping, PhD Thesis, University of Zurich.
CECCONI, A., AND GALANDA, M., 2002, Adaptive Zooming in Web Cartography. In Computer Graphics Forum, 214, 787-799.
CHALMERS, D., SLOMAN, M., DULAY, N., 2001, Map Adaptation for Users of Mobile Systems, In Proc 10th Int. Conf. on World Wide Web, Hong Kong, pp. 735 - 744
CRAMPTON, J. W., 2002, Interactivity Types for Geographic Visualization, Cartography and Geographic Information Science, 29(2), pp. 85-98.
CUTHBERT, A., 1998, User Interaction with Geospatial Data, In OpenGIS® Project Document 98-060.
DEEGREE, 2004, Deegree - Building blocks for spatial data infrastructures, http://deegree.sourceforge.net/ (2004 accessed)
EDWARDES. A. AND BURGHARDT, D. and WEIBEL, R., 2005, Portrayal and Generalisation of Point Maps for Mobile Information Services, In Map-based mobile services – Theories, Methods and Implementations, L. Meng and A. Zipf and T. Reichenbacher (eds), Springer-Verlag:Berlin, Heidelberg, pp 11-28.
EDWARDES, A., D. BURGHARDT, M. BOBZIEN, L. HARRIE, L. LEHTO, T. REICHENBACHER, M. SESTER, AND R.WEIBEL, 2003, Map Generalisation Technology: Addressing the need for a common research Platform, In Proc 21st Int. Cartographic Conf. (ICC), Durban, South Africa, pp. 170-180.
EDWARDES, A., 2004, Modelling space for the generalisation of point maps, In Proc of Geographical Information Systems Research – UK (GISRUK 2004), Norwich UK.
EDWARDES, A., AND MACKANESS, W. A., 1999, Modelling Knowledge for Generalisation of Categorical Maps. In Innovations in GIS 7: GeoComputation Ch. 12.
EDWARDES, A., BURGHARDT, D., AND WEIBEL, R., 2003, WebPark – location based services for species search in recreation area. In Proc 21st Int. Cartographic Conf. (ICC), Durban, South Africa., 2003 (Proceedings on CD-ROM).
ELIAS, B., 2002, Automatic derivation of location maps, In Proc. ISPRS Ottawa, CanadaVol. 34, Pt. 4
FES, 2004, OpenGIS® Filter Encoding Implementation Specification, http://www.opengis.org/techno/implementation.htm (2003 accessed)
FITZKE, J., GREVE, K., MÜLLER, M., POTH, A., 2004, Building SDIs with Free Software – the deegree project, In Proc. 7th Conf. Global Spatial Data Infrastructure (Bangalore, India) (online at http://www.lat-lon.de/download/gsdi-7_full_paper_fitzke_greve_mueller_poth_2004-02-06.pdf - accessed 2004)
FOLEY, J.D., VAN DAM, A., FEINER, S.K., HUGHES, J.F. (1996). Computer Graphics: Principles and Practice, Second edition in C. Reading, MA: Addison-Wesley.
FUHRMANN, S. AND W. KUHN, 1999, Interface Design Issues For Interactive Animated Maps, Proceedings of 19th (ICA/ACI) Int. Cartographic Conf, Ottawa Canada
GAMMA, E. HELM, R., JOHNSON, R.,VLISSIDES, J., 1995, Design Patterns. Addison-Wesley Publishing
GBEI E., I. COSMA, B. MOULIN, N. JABEUR, 2003, Modelling SVG and GML data for the Cartographic Generalisation and the Multiple Representation on the web, In Proc. SVG Open 2003, Vancouver.
GEOAPI, 2004, GeoApi Project, http://geoapi.sourceforge.net/ (2004 accessed)
GEOSERVER, 2004, The Geoserver project – the open internet gateway for geographic data, http://geoserver.sourceforge.net/html/index.php (2004 accessed)
GEOTOOLS, 2004, Geotools – the free open source java GIS mapping toolkit, http://www.geotools.org/ (2004 accessed)
GML, 2004, OpenGIS® Geography Markup Language (GML) Implementation Specification http://www.opengis.org/techno/implementation.htm (2004 accessed)
GO-1, 2004, OpenGIS® Geographic Objects initiative, http://ip.opengis.org/go1 (2004 accessed)
HAMPE, M., ANDERS K.-H., SESTER, M., 2003, MRDB Applications for data revision and real-time generalisation. In Proc. 21st Int. Cartographic Conf. (ICC), Durban, South Africa, pp. 192-202, CD-ROM.
HARDY, P., HAYLES, M. AND REVELL, P., 2003, Clarity - a new environment for generalisation using agents, java, xml and topology. In Proc. 5th Workshop on Progress in Automated Map Generalisation, Paris, France.

HARRIE, L. AND JOHANSSON, M., 2003, Real-time data generalisation and integration using Java, In Geoforum Perspektiv, February, 2003, pp. 29-34.
HARRIE, L. SARJAKOSKI, T., LEHTO, L., 2002, A variable-scale map for small-display cartography, Proc of the Joint Int. Symp. on 'GeoSpatial Theory, Processing and Applications' (ISPRS/Commission IV, SDH2002), Ottawa, Canada
HARRIE, L., AND SARJAKOSKI, L. T., 2002, Simultaneous Graphic Generalisation of Vector Data Sets. GeoInformatica, Vol. 6, No. 3, pp. 233-261.
HJELM, J., 2002, Creating Location Services for the Wireless Web: Professional Developer's Guide, (New York: Wiley).
ICA, 2005, ICA Commission on Generalisation and Multiple representation: R&D resources http://www.geo.unizh.ch/ICA/docs/coding/coding.html (accessed 2005)
ILLERT, A. and AFFLERBACH, S., 2004, Global schema specification. GiMoDig-project, IST-2000-30090, Deliverable D5.3.1, Public EC report, 35 pgs., http://gimodig.fgi.fi/deliverables.php (accessed 2007)
LEHTO, L. AND T. KILPELÄINEN,., 2001, Generalizing XML-encoded Spatial Data on the Web. Proc. 20th Int. Cartographic Conf., Beijing, China, Vol. 4, pp. 2390-2396.
LEHTO, L., AND T. KILPELÄINEN, 2000, Real-Time Generalisation of Geodata on the Web., In Int. Archives of Photogrammetry and Remote Sensing, Amsterdam, Vol. XXXIII, Pt. B4, pp. 559-566.
MENG, L., 2003, Missing theories and methods in digital cartography, In Proc. 21st Int. Cartographic Conf. (ICC), Durban, South Africa., 2003
MUSTIERE, S., 2001, Apprentissage automatique pour la generalisation cartographique, Ph.D. thesis, Université Pierre et Marie Curie (Paris 6).
NEUN, M., R. WEIBEL AND D. BURGHARDT, 2004, Data Enrichment for Adaptive Generalisation. ICA Workshop on Generalisation and Multiple representation (Leicester)
NIVALA, A-M., SARJAKOSKI, L.T., JAKOBSSON, A., AND E. KAASINEN, 2003, Usability Evaluation of Topographic Maps in Mobile Devices, In Proc. 21st Int. Cartographic Conf. (ICC), Durban, South Africa, pp. 1903-1913.
OGC, 1999, The OpenGIS™ Abstract Specification, Topic 14: Semantics and Information Communities http://www.opengis.org (2004 accessed)
OGC, 2003, OpenGIS™ Reference Model, Version 0.1.2. Abstract Specification OGC 03-040. http://www.opengis.org (2004 accessed)
OGC, 2004, OpenGeospatialConsortium Inc. http://www.opengis.org (2004 accessed)
OSI 2004 "Open Source Initiative OSI" http://www.opensource.org (accessed 2004)
PETZOLD, I., GRÖGER, G., AND PLÜMER, L., 2003, Fast Screen Map Labeling - Data-Structures and Algorithms, Proc. 21st Int Cartographic Conf. (ICC), Durban, South Africa, CD-ROM.
RAINSFORD, D. AND MACKANESS, W.A., 2002. Template Matching in Support of Generalisation of Rural Buildings. In: D. Richardson and P.v. Oosterom (Editors), Advances in Spatial Data Handling 10th International Symposium on Spatial Data Handling.. Springer, Berlin, pp. 137-152.
RAPER, J. DYKES, J. WOOD, J. MOUNTAIN, D. KRAUSE, A. RHIND, D., 2002, A framework for evaluating geographical information, Journal of information science 2002 28 (2) pp.39-50
RAPPO, A., CECCONI, A. AND BURGHARDT, D., 2004, Fischaugenprojektionen für generalisierte Kartendarstellungen auf kleinen Bildschirmen, Kirschbaum Verlag, Kartographische Nachrichten, Heft 2.
REGNAULD, N., 1996, Recognition of Building Clusters for Generalisation, In Proc 7th Spatial Data Handling Symp., Delft, Netherlands, pp. 185-198.
REICHENBACHER, T., 2002, SVG for adaptive visualisations in mobile situations, In Proc. SVG Open, 15-17. July 2002 (CD-ROM), Zürich
REICHENBACHER, T., 2003, Adaptive Methods for Mobile Cartography, In Proc. 21st Int. Cartographic Conf, Durban 2003, CD-ROM.
REICHENBACHER, T., 2004, Mobile Cartography – Adaptive Visualisation of Geographic Information on Mobile Devices, Ph.D Thesis (Technical University of Munich) ISBN 3-89963-048-3
RUAS, A., 1999, Modele de généralisation de données géographiques a base decontraintes et d'autonomie Ph.D. thesis, Université de Marne la Vallée.
RUAS, A., PLAZANET, C., 1996., Strategies for automated generalisation, In Proc 7th Int. Symp. on Spatial Data Handling. pp 6.1-6.17.
SARJAKOSKI T, SESTER M, SARJAKOSKI T, HARRIE L, HAMPE M, LEHTO L, KOIVULA T, 2005, Web Generalisation Service in GiMoDig – Towards a Standardised Service for Real-Time

Generalisation, In: Toppen, F., and Painho, M., (eds.), Conference Proceedings of the 8th AGILE Conference on Geographic Information Science, Estoril, Portugal, May 26–28, pp. 509–518.

SESTER, M., L. T. SARJAKOSKI, L. HARRIE, M. HAMPE, T. KOIVULA, T. SARJAKOSKI, L. LEHTO, B. ELIAS, A.-M. NIVALA, AND H. STIGMAR., 2004, Real-time Generalisation and Multiple Representation in the GiMoDig Mobile Service. GiMoDig-project, IST-2000-30090, Deliverables D7.1.1*, D7.2.1* and D7.3.1, Public EC report, 151 pgs. http://gimodig.fgi.fi/deliverables.php (accessed 01/2005)

SLD, 2004, OpenGIS® Styled Layer Descriptor Implementation Specification http://www.opengis.org/techno/implementation.htm (2003 accessed)

SOAP, 2003, SOAP Version 1.2 Part 1: Messaging Framework. http://www.w3.org/TR/soap (accessed 01/2005)

SONDHEIM, M., GARDELS, K., AND BUEHLER, K. (1999) GIS interoperability in Geographical Information Systems: Principles, Techniques, Management and Applications, 2nd Edition, Paul A. Longley, Michael F. Goodchild, David J. Maguire, and David W. Rhind (Eds). John Wiley & Sons, Ch. 24.

SVG, 2004, Scalable Vector Graphics (SVG), http://www.w3.org/Graphics/SVG/ (2004 accessed)

TAKAGI, S., KOBAYASHI, A., TANAKA, T, 2003, Activities for realization of interoperability of location based services using SVG, Proc. SVG Open 2003, Vancouver, Canada.

UDDI, 2004, Universal Description, Discovery, and Integration. http://www.uddi.org/ (accessed 01/2005)

VCKOVSKI, A., 1998, Interoperable and Distributed Processing in GIS. (London:Taylor & Francis)

VIVID SOLUTIONS, 2004, JTS Topology Suite and JTS Conflation Suite, http://www.vividsolutions.com/ (2004 accessed)

W3C, 2004, World Wide Web Consortium, http://www.w3.org/ (2004 accessed)

WARE, J. M. AND JONES, C. B., 1998, Conflict Reduction in Map Generalisation using iterative improvement, GeoInformatica, 2:4 383-407

WEBPARK, 2004, Webpark - Geographically relevant information for mobile users in protected areas, http://www.webparkservices.info/ (2004 accessed)

WEIBEL, R. AND DUTTON, G., 1998, Constraint-based Automated Map Generalisation, In Proc 8th Int. Symp. on Spatial Data Handling (SDH), pp. 214-224.

WEIBEL, R. AND DUTTON, G., 1999, Generalising spatial data and dealing with multiple representations, in Geographical Information Systems: Principles, Techniques, Management and Applications, 2nd Edition, Paul A. Longley, Michael F. Goodchild, David J. Maguire, and David W. Rhind (Eds). John Wiley & Sons, Ch. 10.

WEIBEL, R., 1997, A Topology of Constraints to Line Simplification. In Advances in GIS Research II, M. J.Kraak and M. Molenaar (eds), (London: Taylor & Francis) pp. 533-546.

WFS, 2004, OpenGIS® Web Feature Service Interfaces Implementation Specification http://www.opengis.org/techno/implementation.htm (2004 accessed)

WMS, 2004, OpenGIS® Web Map Service Interfaces Implementation Specification, http://www.opengis.org/techno/implementation.htm (2004 accessed)

WSA, 2004, Web Services Architecture. http://www.w3.org/TR/ws-arch/ (accessed 01/2005)

WSDL 2001, Web Services Description Language (WSDL) 1.1. http://www.w3.org/TR/wsdl (accessed 01/2005)

ZIPF, A. AND RICHTER, K.F., 2002, Using Focus Maps to Ease Map Reading. Developing Smart Applications for Mobile Devices, Künstliche Intelligenz (KI). Sonderheft Spatial Cognition. 04/2002. pp. 35-37.

Research Paper 2

Burghardt, D., M. Neun and R. Weibel (2005): Generalization Services on the Web - Classification and an Initial Prototype Implementation. *Cartography and Geographic Information Science*, Volume 32, Number 4, October 2005, pp. 257-268.

Generalization Services on the Web – A Classification and an Initial Prototype Implementation

Dirk Burghardt, Moritz Neun and Robert Weibel

GIS Division, Department of Geography, University of Zurich
Winterthurerstrasse 190, 8057 Zurich (Switzerland), Fax: +41-1-635 6848
Email: {burg,neun,weibel}@geo.unizh.ch

Abstract Much progress been made in the field of web based cartography through standards developed by the Open Geospatial Consortium (OGC). While automated access and presentation of cartographic data are defined, services for automated generalization are not yet standardized. This paper aims to show advantages of applying the service concept to generalization and suggests several classification schemas of Generalization Services at different levels of granularity. There follows a detailed explanation of a real implemented Generalization Service. It is shown how software developers can make their generalization functionality available as a service and how these services can be accessed dynamically. For the implementation, the open source Java Unified Mapping Platform (JUMP) was extended to work as a framework for generalization. Generalization Services could be used in different application scenarios, for instance as a middleware component between a Web Feature Service (WFS) and a Web Map Service (WMS), to allow on-the-fly adaptive zooming, or as an interactive Generalization Services for the production of topographic maps by national mapping agencies (NMA). They may also allow the development of a common research platform, where researchers would have access to a common generalization framework.

1 Motivation

In recent years cartography has evolved in a new direction with the application of web technologies and services. As a result of this process maps in the web context are generally are generated on-demand and on-the-fly, containing more specific and tailor-made information. This contrasts with the traditional way of map making which focused on the production of static maps that were designed in advance for general purpose usage (with the prototypical example of topographic maps). Web cartography can benefit from the standards developed by the Open Geospatial Consortium (OGC) to implement web services for feature access (WFS) and web mapping (WMS). Obviously, however, the demand for dynamic web map generation has also lead to increased requirements on automated map generalization procedures. Not surprisingly therefore, the OGC has also expressed interest into "feature generalization services" as part of their OGC Web Services (OWS) initiative (OGC and ISO, 2002). However, the specification process for such services has not experienced much progress so far. From another end, the map generalization research community has started to develop an interest into Generalization Services as well, driven by the desire to develop a common open research platform that would allow testing and sharing of generalization algorithms (Badard and Braun, 2003; Edwardes et. al, 2003). This has been evidenced through discussions at the meetings of the ICA Commission on Map Generalization and Multiple Representation in Paris 2003 and Leicester 2004.

There are several advantages to using Generalization Services in a collaborative and distributed research environment as well as for on-demand map production. First of all, the platform independence makes the development independent from the operating system and the hardware used, which also offers application in a mobile context. Secondly, the service can be integrated in any software platforms, such as web browsers, GIS or map production software. Last, the service can be accessed over the internet or on the local machine, the latter however only if the code of the underlying generalization operations is made available.

The objectives of the proposed paper are two-fold: 1) to present a classification of Generalization Services, and 2) to report on a prototype implementations of sample Generalization Services, as well as on experiments that were carried out to assess the feasibility of the approach.

2 Web Services for Spatial Applications and Generalization

2.1 Web Services

In order to enable computers directly to access distributed data and to use services, the concept of Web Services has been introduced. Software programs – frankly computers – can read web pages (mostly in HTML), but they can't understand them. Human beings are able to read a web page and find the link or the button to click on. To enable computers to do this without the help of a human user, *Web Services* make use of machine-processable interface descriptors and of a standardized language:

> *A Web Service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web Service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards. [W3C (2004)]*

The usage of such a Web Service can be either fully automatic within e.g. GIS applications or the Web Service could be called from the application upon the demand of the user.

2.2 Spatial Web Services

Human-computer Interaction Service
Most Spatial Web Services are understood as data delivery services. Usually they connect to a geographic database and retrieve information from a database upon request. Examples are catalog and gazetteer services for data search or Web Mapping Services (WMS) for the visualization of spatial information (Fitzke et. al, 2004). Common to this kind of services is that they are designed for end use by humans. Therefore this category is commonly called human-computer interaction services. The approach represents the view of the Open Geospatial Consortium who have a primary focus on the end user. The aim is to simplify the exploitation of spatial data and the access to geo-information by the user at the end of a chain of spatial services. Very often Web-based Services that are connected to a database can also perform operations upon the data. Examples of such services include a Web Feature Service (WFS), Web Coverage Service (WCS) or a route planning service.

Computer-computer Interaction Service
A second main category is pure processing services which receive data and parameters from the requesting clients and perform operations on them. Possible reasons for this type of services include, for example, the availability of the algorithm only on the server due to licensing or platform incompatibilities, or also the need of faster calculation power on a super-computer. This service category reflects the original purpose of Web Services as given in the above definition, which includes the interoperable interaction between distributed applications, the prerequisite being a complete automation of processes.

2.3 Application of Service Concepts to Generalization

In general two concepts of Generalization Services seem to have a promising range of application. The one and potentially most widely used service would be the generalization or better the adaptive zooming in Web Map Services. Providing a zoom function is classically the domain of Multi-Resolution Databases (MRDB). A Generalization Service could be introduced as an add-on to a WMS or as a **middleware** between a WMS and the client which produces the desired resolution out of the available data. This kind of service belongs to the second category of computer-computer interaction service, which requires fully automated solutions.

The other promising service would be more interactive Generalization Services for GIS users and for map production in organizations such as national mapping agencies (NMA). In this case the Generalization Service would provide its functionality and its calculation power to the service subscribers. It also fulfills the requirements of a **common research platform** (Edwardes and Burghardt, 2005), where the researchers want to have access to a common generalization framework. Advantages of such a **standalone service** would include, for example, the ability to provide specialized or novel algorithms to the research community without everybody having to adapt their systems to the specific needs. Furthermore, the possibility exists to write specific algorithms for special computer architectures e.g. clusters, grids or other parallel processing systems and offering this service to the subscribers.

Figure 1 illustrates the two types of Generalization Services. In the case of zooming only configuration

parameters such as the resolution and the bounding box are sent from the client to the Generalization Service. In the case of a specialized Generalization Service the data would also have to be sent. In the first case the service knows exactly the type and structure of the data (Lehto and Sarjakoski, 2004; Illert and Afflerbach, 2004). In the second case this is a major problem to handle. Only standardized and valid data can be handled by the Generalization Service.

Additionally, the kind of interaction with such services differs. While the Generalization Service runs completely automatically in the first instance, the user has (i.e. wants to have) more control in the second, interactive scenario. These different usage scenarios will be outlined in detail in section 4.

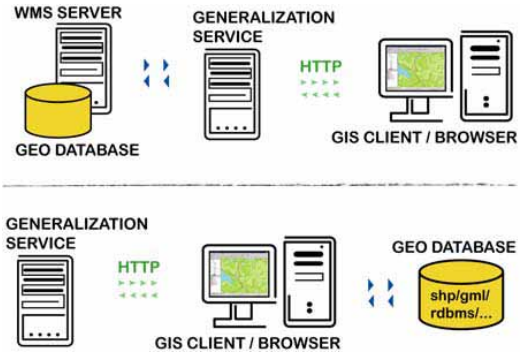


Figure 1: Two types of Generalization Service: As a middleware (above) or as research platform (below).

3 Characteristics for Generalization Services on the Web

There are several characteristics for Generalization Services using general concepts of Web Services. First, the conceptual and implementation characteristics will be outlined. It is shown what a service oriented architecture is and which techniques are available to offer Generalization Services to service consumers, such as human users or cartographic software applications.

3.1 Service-Concept and Component Architecture

A **service** is an abstract **resource** that represents a capability of performing various tasks. This abstract service has to be realized by a concrete agent. Different services can be connected with a request based communication. This **service oriented architecture** approach offers the resources to other users in a network as independent services. Service access is achieved through a standardized approach. These loosely coupled services offer more flexibility than other system architectures. In essence, service-oriented architectures represent collections of services communicating with each other. Communication can be either simple data passing or it could also be two or more services jointly coordinating some activity. A service represents the endpoint of a connection. A service has an underlying computer system that manages the client-server connection. Service invocation is done by a **service request** to the invokable interface of the service. Upon successful operation, the service provider returns a **service response**. Errors have to be handled with **service exceptions**. These interactions are independent and interfaces and protocols yield the underlying infrastructure. The **Web Services** technology offers these capabilities for service communication and interfacing.

Services can be used by higher level services and can themselves use the functionality of others. This is achieved through the ***n*-tier distribution** capabilities of Web Services. This structure implies the client/server program model. The processing of a specific application occurs over *n* machines across a network. These tiers usually include a data tier, business logic tier, and a presentation tier. A given machine will perform the specific tasks of a tier. Sometimes multiple tiers can also reside on one machine but this usually is only used for testing purposes. In that case the remote architecture remains unchanged. *N*-tier applications have not only the advantages of distributing computing but additionally any one tier can run on an appropriate processor or operating system platform and every tier can be updated independently.

The **component architecture** emerges from object-oriented and internet technologies. The systems components in a component-based architecture have generic interfaces through which they advertise their functionalities. This enables the dynamic loading of the components. Components, software objects encapsulating a set of functionalities, interact with other components. Every component has an interface which conforms to a defined architecture.

3.2 Conceptual and Technical Characteristics for Web Services in General

Every Web Service is a resource. The Web Service architecture implements a service oriented architecture using web technologies. The following main characteristics can be defined for all Web Services:

- platform independence
- service registry
- Web API - interface
- loosely coupled communication

The **platform independence** of a Web Service is a major advantage. This feature, also referred to as interoperability, enables a Web Service to be really distributed over many different platforms and distinguishes Web Services from other technologies such as CORBA or Java RMI. When browsing the web and accessing services with a browser users expect to be able to see and use a web page on any platform or operating system. They don't want to care whether they are using a Windows, Macintosh or UNIX/Linux machine. Web pages use a common set of standardized protocols and languages. To achieve this interoperability equally for Web Services already existing standards have been chosen. Web protocols ensure easy integration of heterogeneous environments. The **protocol HTTP** and the **language XML** are available for every major platform.

In order to announce the availability of a Web Service and to find Web Services a registry is needed. The **"publish-find-bind"** principle describes this functionality. Web Services are registered (*publish*) and can be located through a Web Service **registry**. Service consumers can *find* suitable Web Services through a registry. These service consumers may be humans or other applications. The registry offers a single point access which then gives the exact service endpoint of the service's implementation to the service consumer (*bind*).

The **Web API** is the **interface** of a service which can be called from other programs. This interface is a standards based application-to-application programming interface which can be invoked from nearly any type of program. In order to enable programs to bind an interface automatically, the capabilities of an interface are shown through self-description. This usually is achieved through an interface description language (i.e. WSDL). The binding of a Web Service specifies the protocol and data format used for transmitting messages to the invoked interface.

Web Services are **loosely coupled**. The systems pass XML messages, usually over the HTTP protocol, to each other via their Web API. The Web API interface is the abstraction layer which yields the real communication and makes the connections stable and flexible. The protocol HTTP and the language XML can transport nearly any type of data. For instance binary arrays can be converted into ASCII and then packed into an XML document. This approach makes the Web Service concept very open and usable for many different purposes.

3.3 Advanced Characteristics for Generalization Services

Depending on the usage concept, either as a standalone service (e.g. research platform) or as middleware, different requirements of service registry and service invocation exist. The middleware concept often needs no registry because mostly it forms a combined system together with the data source (e.g. WMS). So, in this case the middleware would form some kind of service endpoint for the data source. In such a case the Generalization Service is bundled with the data access. The success of Generalization Services as a common research platform or standalone Generalization Service, however, is very much dependent on the existence of some kind of **service registry** (cf. section 5.1). The service registry has to be the single point access to all available services. These "Yellow Pages" for Generalization Services must know where the interface description and the service endpoints of the desired service can be found. The concrete architecture of such a registry will be described later in this paper.

Like the service registry, the **service invocation** also depends on the usage concept. A middleware concept for the access to generalized data would only need to transfer parameters to the service while a research platform or a standalone Generalization Service also needs functionality to upload data. All these concepts can also have different ways of interaction. This can be the classical human-computer interaction which is based on forms (in web pages) or on application plugins. The computer-computer interaction could also be fully transparent to the user or be hidden in a cartographic system without any control or interaction of the user.

In the case of a Generalization Service which supports the upload and treatment of the individual users' data there are special requirements for the **interface** and the **encoding of geo-features** with their geometry. In a form-based web page environment with full human interaction transfer formats such as Shapefiles or

GML would be suitable. But they have to respect the format needed by the service. When using a plug-in in a cartographic application, the communication logic of the plug-in has to translate the application's internal concept of geo-features to a common format which can be understood by the service (cf. section 5.2).

Generalization of more than one feature and complex Generalization Services need two additional features which are not built into Web Services. These features are the capability to maintain the program's **execution state** and the support for **atomic transactions**. Maintaining the state of the execution of an algorithm is very important in the case when client interaction with the service at run time is desired. Once the client has uploaded the initial parameters (and possibly data) the service starts the execution of the algorithm. If the algorithm needs additional data, parameters or user input, a response with the corresponding request is sent back to the client. While this communication takes place the service has to maintain the already entered information, data and the already calculated changes. As the Web Services are based on loosely coupled state-less communication this has to be accomplished additionally. Technologies for this purpose such as session management via session IDs or cookies are usually already built into many Web Servers. The notion of transactions is fully compatible with the transaction concept in database systems. While executing a complex algorithm on multiple features in one data layer the data set is only changed if no error occurs. Otherwise a roll-back is done and no changes are committed. This feature is important to maintain the original data set's consistency in generalization.

The maintenance of state and the concept of transactions become extremely important when advanced **service control** and especially the **chaining** of several services is desired. The chaining of different services with their algorithms is one important step towards a "generalizeMap()" operation which generalizes an entire map and delivers a more or less ready-to-use product. For the use in middleware systems with the objective of fast display of generalized map objects the steps of the processing chain would be fully opaque. The quality in this case is less important than speed and stability. In the case of map production and research service chaining should also include interaction with the user in order to obtain high quality results. In this case usually the usage concept would be implemented with a plug-in because a form-based solution (e.g. in a browser) does not offer the flexibility and the support for long-time connections.

4 Categories for Generalization Services

4.1 Categorization Based on OGC/ISO Architecture Model (02-025)

As discussed above the distinction of spatial services according to the involvement of humans has lead to the categories of *human-computer* and *computer-computer interaction services*. The OGC/ISO architecture model refines these two main service categories with the following subdivision:

- Processing services
- Model/information management services
- Workflow/task management services
- Human interaction services
- Communication services
- System management services

Services belonging to these different categories can be applied usefully to the generalization process, as we will now show. *Processing services* encompass services that perform large-scale computations as they are needed when a generalization process is carried out fully automatically, for instance, on a complete map image or on generalization sub-tasks which can be completely separated from each other, such as text placement. Typically processing services do not include capabilities for providing persistent storage of data. With regard to our application scenarios processing services are needed to implement on-the-fly generalization, such as adaptive zooming. Between data access through a Web Feature Service (WFS) and map presentation with the help of a Web Mapping Service (WMS), the Generalization Service runs completely automatically as a middleware to adapt information to the screen size of an output device, used symbolization and map content. In this context performance is very important, so lower quality of the Generalization Service will be accepted. Lower quality of Generalization Service means only simple selection and simplification operations are carried out in real time, while more time consuming, context dependent generalization operations will be omitted. Lower quality maps usually are sufficient for display on mobile devices where the image loading time is very important for the user's look-and-feel and too many details can not be displayed due to limited screen resolution.

The other application scenarios use the Generalization Service as support for interactive map production and user controlled semi-automated derivation of multiple

representations at smaller scales. The *model and information management services* category from the OGC/ISO architecture model can be seen as collection of these kinds of services, which allow the development, manipulation, and storage of metadata, conceptual schemas, and datasets. For this type of application the main control lies on the user side. *Workflow services* can help to define, invoke, status and control service chaining. *Human interaction services* allow interaction during the generalization process, for instance, to decide which object classes have to be generalized or selected and to parameterize Generalization Service operators. The OGC/ISO architecture model suggests two additional categories, one for *communication services* to encode and transfer data across networks and one for *system management services* which include, for instance, management of user access privileges. All these kinds of services can be advantageously combined in a generalization research platform.

4.2 Default and Advanced Generalization Service Category Based on GML Specification of Open Geospatial Consortium

The second way of categorizing Generalization Services is based on the OpenGIS® Geography Markup Language (GML) Implementation Specification, which differentiates between GML core schema elements and GML application schema elements. Default Generalization Services as one category deal with GML core schema elements and could be used with any GML dataset. In many cases, for example when simplifying single area objects (e.g. the geometry of a building) or line objects (e.g. the geometry of a river or street) application requirements may be simple and the default generalization behaviors could suffice to meet those requirements. Advanced Generalization Services as the second category encompass services for GML application schema elements. Application schema elements allow additionally the modeling of objects by means of attributes and object compositions.

Default Generalization Services contain generalization functionality which operates on the micro level. This means that generalization operations are carried out on single objects, while context dependency will not be considered. According to Ruas (2000) micro objects generalize themselves or react to orders for contextual operations given by (superordinate) meso objects. Using the typology of McMaster and Shea (1992) the following operators can be applied on single objects: simplification, smoothing, geometry type change, collapse, enhancement and selection based on geometry. Default Generalization Services allow the reduction of resolution by means of a single geometrical operation (e.g. line simplification) and simplified modeling (e.g. centerline representation of roads and streets).

Advanced Generalization Services consider also attributes, symbolization and spatial context through neighboring objects. Hence, generalization operations have to deal with groups of objects, so called meso objects. Meso objects (or meso agents) generalize themselves when they perform contextual operations (Ruas 2000). Advanced Generalization Services allow to implement the remaining, contextual generalization operators of the typology by McMaster and Shea (1992), including context-dependent selection, aggregation, typification, exaggeration and displacement of map objects, taking into account both the geometry as well as semantics and attributes of the objects involved.

4.3 Hierarchical Categorization

Extending the idea of default and advanced Generalization Services, a hierarchical breakdown can be made which distinguishes between the following categories

1. Generalization support service (e.g. services for buffering or for creating a topological data structure, a skeleton or a constrained Delaunay triangulation)
2. Generalization operator services (e.g. services for line simplification, line displacement, area aggregation)
3. Generalization process services (e.g. services for automated orchestration, services for evaluation of generalization results)

The first category contains services that should support the generalization process and the generalization operators. Therefore this category represents the lowest level of this hierarchical categorization. Examples are services for creating a topological data structure or services for creating a constrained Delaunay triangulation. The results of such a service is additional cartographic information which can be optionally stored also in a Multi-Resolution Database (MRDB). These kinds of services can be seen as enriching data with respect to the automated generalization process. The main goal of such services is to make information explicit, representing common structural properties such as neighborhood or proximity relations and alignments which can be usefully exploited by generalization operations (Neun et al. 2004).

The second service category delivers the functionality of standalone generalization operators. Several typologies of such generalization operators are suggested for instance by McMaster and Shea (1992). Examples are services for simplification, smoothing, aggregation, amalgamation, merging, collapse, refinement, exaggeration, enhancement and displacement. These generalization operator services can be further subdivided for point, line and area objects and specialized depending on object classes. It is obvious that rivers, borders and railway lines have to be generalized in a different way, despite the fact that all are line objects. The generalization operators of this second service category are offered in an interactive mode, with the user selecting appropriate generalization operators/algorithms as well as setting the control parameters of the algorithms.

The third hierarchy level of generalization process services uses services from lower categories and encompasses services which allow the control and orchestration of generalization operators. Examples are the meso agents as described for the advanced Generalization Service category (Ruas 2000). Automated control of the generalization process presently receives ample attention as a research topic. Besides agent-based modeling also combinatorial and continuous optimization approaches are proposed in the literature. Simulated annealing (Ware et al., 2003) as a combinatorial optimization approach allows the selection of generalization operations controlled by assigning costs to each operation. Continuous optimization approaches include the finite element method (Højholt, 2000), snakes or elastic beams (Burghardt and Meier, 1997; Bader, 2001; Galanda and Weibel, 2003) and least-squares adjustment (Harrie, 1999; Sester 2000). The latter methods are primarily used to control generalization operations of continuous nature, such as displacement or smoothing. All approaches mentioned so far, however, are still quite a way from a perfect modeling of the overall map generalization process. In the meantime, an interim mechanism for controlling the generalization process is provided by the three architecture patterns for service chaining in the OpenGIS® Web Services Architecture (OGC, 03-025) which can be used depending on purpose and map complexity:

- User defined (transparent) chaining: the human user entirely manages the workflow. For complex generalization processes for which no adequate process modeling exists yet.
- Workflow-managed (translucent) chaining: the human user invokes a Workflow Management service that controls the chain and the user is aware of the individual services. For medium-complexity generalization processes that can be specified as workflows.
- Aggregate service (opaque): the human user invokes a service that carries out the chain, with the user having no awareness of the individual services. For relatively simple, sequential generalization processes.

5 Implementation

The prototype implementations of Generalization Services reported here include examples of rather simple services (e.g. Douglas-Peucker line simplification, building simplification) where spatial context is not considered. The objective is to show the feasibility of the service-based approach and to describe the minimum set of components needed to run the Generalization Services over the internet. For the implementation of the Generalization Services we use an open source framework for mapping application called JUMP (http://www.jump-project.org) that delivers standard functionality for reading and writing files (Shape, GML), as well as modifying and visualizing cartographic data. JUMP is written in Java, which allows easy application service provision (ASP) over the internet. The usage of the framework's functions is enabled by including the JUMP libraries in Java servlets which are running in a servlet engine such as TOMCAT. These servlets contain algorithms with the generalization logic or controls for external generalization libraries, respectively.

5.1 Registry for Generalization Services

The central point for accessing and publishing Generalization Services is the registry. Comparable to "yellow pages" the registry has to be used when looking for available generalization functionality. It would make sense for some large independent organization such as the ICA Commission on Map Generalization and Multiple Representation to host such a registry database.

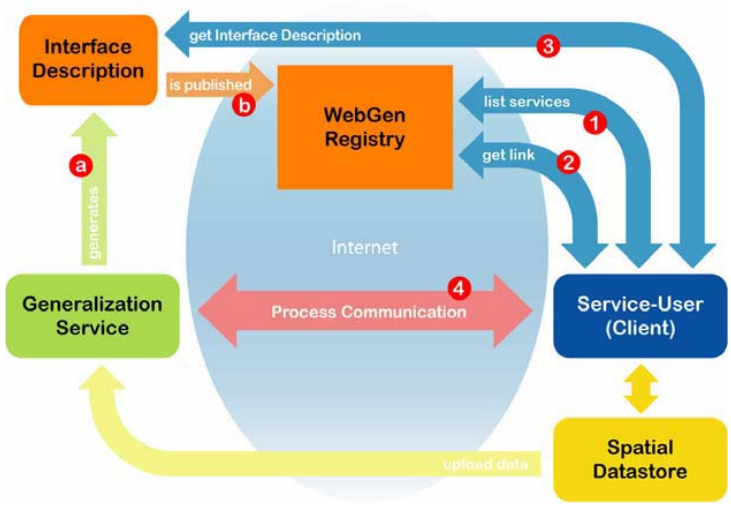


Figure 2: Registry for Generalization Services

To offer a Generalization Service the following steps have to be performed by the service provider, for instance by a research group (cf. Figure 2). The first task is to create (a) an interface description containing the parameters of the Generalization Service and the service endpoint, which consists of a URL address where the service can be accessed. Once the interface description is published (b) in the registry database the community can access the service.

Clients looking for Generalization Services can use the registry to find (1) available services. Selecting a desired Generalization Service the client obtains the link (2) to the interface description of the service. Accessing the interface description (3) the user knows the endpoint of the service, as well as the number, names and types of the parameters. With this information the client plug-in can create automatically a user interface for the selected service, allowing the specification of generalization parameters. Having a central access point is one advantage of a registry. Additionally, the registry automatically identifies service endpoints, so there is no active change needed by the service users if the service provider changes the location of the software on the server. Finally, the process communication (4) is responsible for the transfer of parameters and data between user and service provider.

5.2 Process Communication and Feature Metadata (Encoding and Upload)

The two main service concepts (cf. section 2.3) require different ways of communication for the data input (upload) and the output (download). Figure 3 illustrates three different ways to implement process communication. The concept of a research platform is either implemented as a form-based web page in a browser or as a plug-in for mapping software. The browser upload offers only the file upload via HTTP. The selected file is encoded automatically by the browser and has to be decoded on the server. The browser example offers the possibility to upload and process Shapefiles. As output in the browser the user can download a newly created Shapefile with the result of the generalization algorithm.

The implementation as a plug-in on the client GIS offers the possibility to encode the data directly out of the application. This gives the possibility to choose a better suited format for the transfer. In the example plug-in, the geo-features are encoded in a GML compatible format directly into a SOAP message (Simple Object Access Protocol), then transferred to the server. The use of another format (e.g. a binary format) would also be possible, with the possibilities of SOAP envelopes and e.g. MIME encoding. The output of the server is in the same format as the request and can again be decoded on the client and displayed there.

The concept of a middleware layer between the data source and the user (mostly in conjunction with a browser) introduces a third possibility for getting the data to the Generalization Service. In this case the

client would simply send the URL of the data source (e.g. a WFS) to the Generalization Service as a parameter in the call. The server itself would then access the data source, process the data and send it back to the user (Sester et. al, 2004).

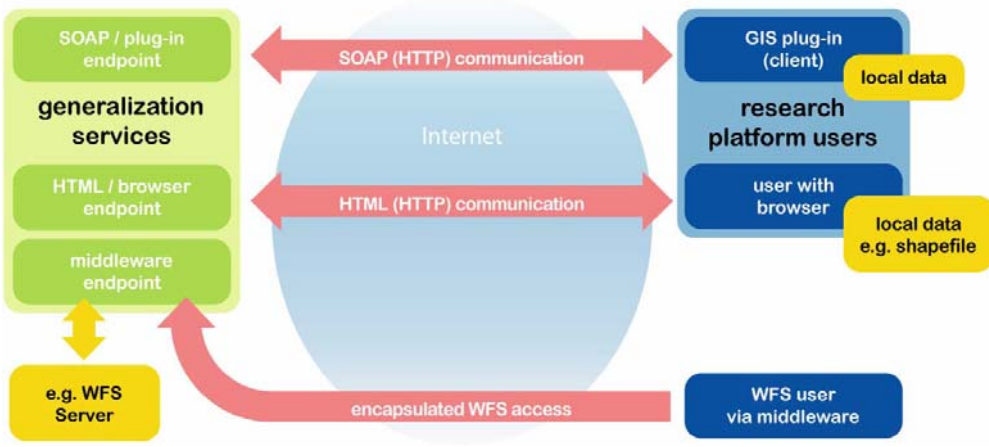


Figure 3: Three ways to implement process communication

The metadata specification for the geo-features is treated in the following way. Every feature can have an arbitrary number of attributes. The number, name and type (spatial or non-spatial) of the attributes needed by the algorithm are specified in the interface description (cf. section 5.3). Generalization data is always geometrically dominated. This means that every feature has at least one geometry plus possibly other attributes. The interface of a road generalization algorithm could for example require a geometry for each road and the road class (highway, major road, etc.). Listing 1 shows an example for such a schema specification. The users of the service would then in their client select the columns with the geometry and the road class in their local data set. For the data transfer the client then automatically assigns the name from the schema to the selected column. So, the server can identify which attributes it can use. Other attributes in a local dataset will simply be ignored not deleted.

```
<schema>
  <attribute>
   <name>geometry</name>
   <type>GEOMETRY</type>
  </attribute>
  <attribute>
   <name>class</name>
   <type>STRING</type>
  </attribute>
</schema>
```

Listing 1: Example feature schema

For the browser upload and the plug-in example the necessary encoder and decoder methods have been implemented in Java. These classes form the communication framework which can be used for the client-server communication.

5.3 Service Invocation and Client Implementation

On the client side both an example for the browser based Generalization Service and a plug-in for the JUMP Unified Mapping Platform have been implemented. Other plug-ins for platforms such as ArcView® are also planned. The client implements an easy to use GUI (Graphical User Interface) for the service user. The browser example works with standard HTML pages in every major browser platform. The user accesses such a service through a start page (user authentication with password can be activated). This start page, containing all available services, is dynamically created with information from the service registry (cf. section 5.1). After selecting a particular Generalization Service the user is presented a new, dynamically created page which allows him/her to enter the parameters for the algorithm and upload a Shapefile from his/her local system.

The JUMP plug-in does mostly the same as the browser solution but it integrates seamlessly into the software so that the user does not have to quit the application and even does not necessarily notice a big difference between using a local or remote algorithm. The plug-in integrates into the JUMP menu bar. The first time after the installation of the plug-in the user has to enter the URL of the registry (cf. section 5.1). With this URL the plug-in automatically looks every time it is started for all available Generalization Services. The result of this query is displayed in a selection list to the user (see Figure 5).

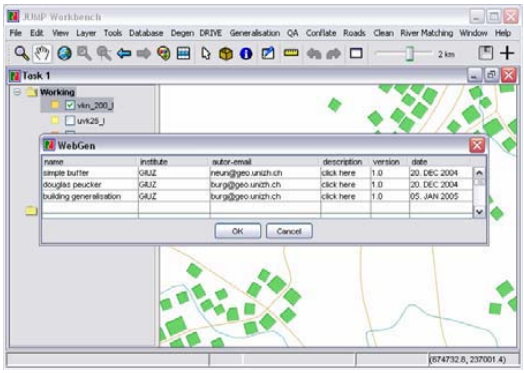


Figure 5: List of available services (from the Generalization Service registry)

From the list of available services the user selects the desired Generalization Service. The user then is presented an entry form for the corresponding algorithm's parameters. The configuration of all those entry forms is dynamically created from the interface description (cf. section 5.1). An example of such a simple interface description for the Building Simplification algorithm is shown in Listing 2. The data format for the interface description is XML. The important parts of the interface description (Listing 2) are highlighted.

```
<?xml version="1.0" encoding="UTF-8"?>
<webgen>
<name>building generalization</name>
<method>buildingGenNew</method>
<endpoint>http://www.geo.unizh.ch:8080/neun/servlet/GenHandlerXML</endpoint>
<description>This algorithm simplifies all buildings in a layer and returns a layer containing the resulting buildings!</description>
<config>
 <layer>
  <schema>
   <attribute>
    <name>GEOMETRY</name>
    <type>GEOMETRY</type>
   </attribute>
  </schema>
 </layer>
 <param>
  <name>min edge length</name>
  <type>DOUBLE</type>
  <description>Minimum Edge Length!</description>
 </param>
</config>
</webgen>
```

Listing 2: The interface description in XML

The entire description is in one XML container. The tag <method> specifies the generalization algorithm which has to be used on the server and has to be passed to the server proxy. The tag <endpoint> specifies the URL of the server proxy. The tag <layer> specifies the minimum schema (metadata) the algorithm needs. In the example (Building Simplification algorithm) only geometries (<name> and <type>) are needed. Other attributes can also be contained in the layer but they will be ignored. For another algorithm like road re-classification e.g. a second attribute indicating the class of the road would be needed. The other parameters which are needed by the algorithm are indicated by the <param> tag. There can be as many parameters as needed. In the example there is only one parameter, the minimal edge length (type DOUBLE), needed.

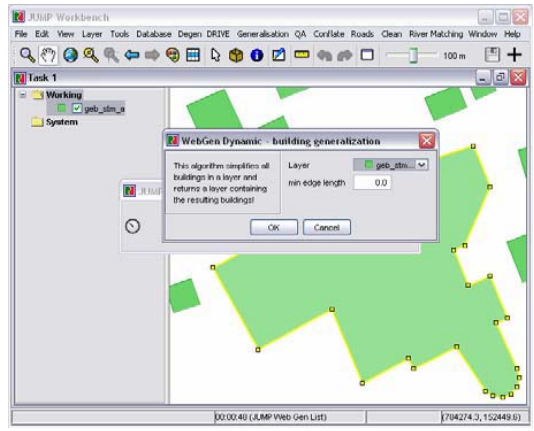


Figure 6: Algorithm interface generated dynamically from an interface description

Figure 6 shows the automatically generated entry form for the Building Simplification algorithm. After select-

ing the layer with the geometries and entering the tolerance, the data is transferred to the server and processed. The resulting geometries are sent back to the client and displayed in a new layer in JUMP.

The implementation example assumes a near real-time execution of the Generalization Service. The client is kept in a blocked wait mode. In this case the generalized data can be delivered immediately back to the client. Execution time is limited by client and server timeout settings and users' patience. For more time consuming operations, a system which informs the user when the service is finished would be possible and preferable. So, during the processing of a request the client would not be blocked and could perform other tasks. This has not been implemented so far, however. For the middleware concept only the real-time execution is of interest, because the user usually wants to see the map very quickly. The (simple) algorithms implemented so far have all performed very fast, so the bottleneck was not the computing time but the network transfer time. In the middleware concept, assuming a fast connection between data source and Generalization Service, this is negligible.

6 Conclusion

In an attempt to stimulate the discussion about Generalization Services in the generalization research community this paper explained the minimal requirements for the client and server side implementation of Generalization Services. It was shown how developers and researchers can make their generalization functionality available by means of an interface description and how possible users of these services can find them through a registry database. As an example simple Generalization Services for line and building simplification were implemented and accessed dynamically from the open source Java Unified Mapping Platform. To show the advantages of Generalization Services two different application scenarios were proposed. The first scenario implements the processing service as a middleware solution in order to realize on-the-fly generalization for tasks such as adaptive zooming. The second application scenario focuses on interactive generalization as support for interactive map production and user controlled semi-automated derivation of multiple representations at smaller scales.

The limitations of our solution are that no symbolization was considered so far and only context independent Generalization Services were implemented. Also, there is no user or session management up to now. This would be needed for longer computations. Further research has to investigate ways of interacting and chaining of Generalization Services, which is related to the yet largely unsolved problem of automated orchestration of generalization operators. To take full advantage of distributed services and grid computation ways have to be found for separating and distributing generalization tasks. The partitioning of a generalization task based on the identification of independent problems (e.g. the generalization of building blocks surrounded by streets) or the subdivision of the map area with solving boundary problems is imaginable.

Acknowledgements

Research reported in this paper was partially funded by the Swiss NSF through grant no. 20-101798, project DEGEN. We are grateful to Alistair Edwardes for helpful comments on the paper.

References

Badard, T., and A. Braun. 2003. OXYGENE: An open framework for the deployment of geographic web services. In: *Proc of the 21st Int. Cartographic Conf.,Durban, South Africa.* (CD-ROM).

Bader, M. 2001. *Energy Minimization Methods for Feature Displacement in Map Generalization*. Doctoral Thesis, Department of Geography, University of Zurich, Switzerland.

Burghardt, D. and S. Meier. 1997. Cartographic Displacement Using the Snakes Concept. In: Förstner, W., and L. Plümer (eds), *Semantic Modeling for the Acquisition of Topografic Information from Images and Maps*, Birkhäuser Verlag, 59-71.

Edwardes, A., D. Burghardt, M. Bobzien, L. Harrie, L. Lehto, T. Reichenbacher, M. Sester and R. Weibel. 2003. Map Generalisation Technology: Addressing the need for a common research platform. In: *Proceedings of 21st International Cartographic Conference. Durban, South Africa.* (CD-ROM)

Edwardes, A. and D. Burghardt. 2005. Experiments to build an open generalisation system. In: W. Mackaness, A. Ruas and T. Sarjakoski (eds), *Challenges in the Portrayal of Geographic Information: Issues of Generalisation and Multi Scale Representation.*.

Fitzke, J., K. Greve, M. Müller, A. Poth. 2004. Building SDIs with Free Software – the deegree project.

In: *Proc. 7th Conf. Global Spatial Data Infrastructure (Bangalore, India).*

Galanda, M. and R. Weibel. 2003. Using an Energy Minimization Technique for Polygon Generalization. *Cartography and Geographic Information Science,* 30(3): 259-275.

GML. 2005. OpenGIS® Geography Markup Language (GML) Implementation Specification http://www.opengis.org/techno/implementation.htm (accessed 01/2005).

Harrie, L. 1999. The Constraint Method for Solving Spatial Conflicts in Cartographic Generalization. *Cartography and Geographic Information Science*, 26(1): 55-69.

He, Hao. 2003. What is Service-Oriented Architecture? http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html

Højholt, P. 2000. Solving Space Conflicts in Map Generalization: Using a Finite Element Method. *Cartography and Geographic Information Science*, 27(1): 65-73.

Illert, A. and S. Afflerbach. 2004. Global schema specification. GiMoDig-project, IST-2000-30090, Deliverable D5.3.1, Public EC report, 35 pgs., http://gimodig.fgi.fi/deliverables.php (accessed 01/2005)

JUMP. 2005. JUMP Pilot Project – Fostering the Development of the JUMP GIS Platform. http:// www.jump-project.org; http://jump-pilot.sourceforge.net/index.php (accessed 01/2005).

Lehto, L., and T. Sarjakoski. 2004. An Open Service Architecture For Mobile Cartographic Applications. In: G. Gartner (ed.), Location Based Services & TeleCartography, *Proceedings of the Symposium 2004*, Vienna University of Technology, January 28-29, 2004, Vienna, 141-145.

McMaster, R., and S. Shea. 1992. *Generalization in Digital Cartography*. Washington D.C: Association of American Geographers.

Neun, M., R. Weibel and D. Burghardt. 2004. Data Enrichment for Adaptive Generalisation. In: *ICA Workshop on Generalisation and Multiple Representation* (Leicester), available from http://ica.ign.fr/

OGC and ISO. 2002. The OpenGIS™ Abstract Specification, Topic 12: OpenGIS Service Architecture, Version 4.3. Abstract Specification OGC 02-112. Also available as ISO/DIS 19119 – Geographic Information Services.

Ruas, A. 2000. The Roles of Meso Objects for Generalisation. In: *Proceedings 9th Symposium on Spatial Data Handling (Beijing, China)*: 3b50-3b63

Sester, M. 2000. Generalization Based on Least-squares Adjustment. In: *International Archives of Photogrammetry and Remote Sensing*, Vol. XXXIII, Part B4, Amsterdam, 931-938.

Sester, M., L. T. Sarjakoski, L. Harrie, M. Hampe, T. Koivula, T. Sarjakoski, L. Lehto, B. Elias, A.-M. Nivala, and H. Stigmar. 2004. Real-time Generalisation and Multiple Representation in the GiMoDig Mobile Service. GiMoDig-project, IST-2000-30090, Deliverables D7.1.1*, D7.2.1* and D7.3.1, Public EC report, 151 pgs. http://gimodig.fgi.fi/deliverables.php (accessed 01/2005)

Vivid Solutions. 2004. JTS Topology Suite and JTS Conflation Suite, http://www.vividsolutions. com/ (accessed 01/2005)

Ware, J.M., C.B. Jones and N. Thomas. 2003. Automated Map Generalization with Multiple Operators: A Simulated Annealing Approach. *International Journal of Geographical Information Science*, 17(8): 743-769.

W3C. 2005. Web Services Architecture. http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/ (accessed 01/2005)

WFS. 2005. OpenGIS® Web Feature Service Interfaces Implementation Specification. http://www.opengis.org/techno/implementation.htm (accessed 01/2005)

WMS. 2005. OpenGIS® Web Map Service Interfaces Implementation Specification. http://www.opengis.org/techno/implementation.htm (accessed 01/2005)

Research Paper 3

Neun M., D. Burghardt and R. Weibel (in press): Web Service Approaches for Providing Enriched Data Structures to Generalisation Operators. Accepted for *International Journal of Geographical Information Science*.

International Journal of Geographical Information Science
Paper will appear in 2007

Web service approaches for providing enriched data structures to generalisation operators

MORITZ NEUN*, DIRK BURGHARDT and ROBERT WEIBEL

University of Zurich, Department of Geography, Winterthurerstrasse 190, CH-8057 Zurich, Switzerland

Web service technologies can be used to establish an interoperable framework between different generalisation systems. In a previous article three categories of generalisation web services have been identified, including support services, operator services and processing services. This paper focuses on the category of support services. In a service-based generalisation system, the purpose of support services is to assist the generalisation process by providing auxiliary measures, procedures and data structures that allow to represent structural cartographic knowledge. The structural knowledge of the spatial and semantical context and the modelling of structural and spatial relationships is critical for the understanding of the role of cartographic features and thus for automated generalisation. Support services should extract and model this knowledge from the raw data and make it available to other generalisation operators. On the one hand the structural knowledge can be expressed by enriching map features with additional geometries or attributes. On the other hand there exist various hierarchical and non-hierarchical relationships between map features, many of which can be represented by graph data structures. After a brief introduction to the interoperable web service framework this paper proposes a taxonomy of generalisation support services and discusses its elements. It is then shown how the complex output of such services can be represented for use with web services and stored in a reusable fashion. Finally, the utilisation of support services is illustrated on four implementation examples of support services that also highlight the interactions with the generalisation operators that use these auxiliary services.

Keywords: automated map generalisation, web services, feature generalisation services, generalisation support services, structural knowledge, supporting data structures, graph data structures, triangulations

1. Introduction

Structural knowledge, that is, the knowledge of the spatial and semantical context of map features and their structural relationships is critical for the understanding of the role of the features represented on a map and thus for automated generalisation (Armstrong, 1991; Ruas and Plazanet 1996; Regnauld, 2005). This supporting knowledge, however, has rarely been coded directly into the commonly available cartographic databases. Hence, algorithms for extracting structural knowledge from the raw cartographic data are required in order to 'enrich' the available cartographic databases (Ruas and Plazanet 1996). Such algorithms – like other generalisation algorithms – are being developed by various research groups, typically on their platform of choice. The problem, then, is that although the generalisation toolbox is seemingly steadily filling up no real progress can be made in research nor in production as long as these tools stay on different platforms (Edwardes et al. 2003). Web services can be used to make this knowledge available in a platform independent way allowing the combination of different algorithms and supporting data structures using a common interface (Edwardes et al. 2007, Burghardt et al. 2005).

In a previous article (Burghardt et al. 2005) we have introduced, for the first time, a comprehensive architecture of generalisation web services and demonstrated its use on an initial implementation of simple generalisation algorithms. As the original paper argues, a complete service-based generalisation system will require three types of services: operator services, support services, and process services. Operator services implement generalisation operators (e.g. simplification, smoothing, aggregation); support services provide supporting measures and data

* Corresponding author. Email: neun@geo.unizh.ch

structures to the operator services; and process services are responsible for workflow control (of operator and support services) and evaluation of the generalisation results. This paper focuses exclusively on support services, attempting to define their role within a service-based generalisation system; proposing a taxonomy of support services that are considered essential for structure recognition in map generalisation; discussing issues of the interoperable storage and exchange of the out put of such support services; and presenting and discussing what we believe to be representative implementation examples. With this, we hope to provide a foundation for the systematic further development of generalisation web services, an example being the WebGen framework (§ 2.4), a service platform that represents a joint effort of several authors residing in different research groups.

Generalisation web services encompass a whole set of different methods and processes in order to deliver generalised maps adapted to constraints such as legibility or user needs. The use of web services in generalisation is not limited to the creation of web maps for display in a web browser. Generalisation services can also be used for the coupling of different generalisation platforms and for the provision of generalisation functionalities as an interoperable toolbox within a map production context.

In Section 2, we will briefly review the main elements of the initial framework for generalisation web services as a foundation of this paper and to provide an embedding for support services used in generalisation. In Section 3, the paper goes on to define and discuss a taxonomy of generalisation support services. In Section 4, it is shown how the complex output of such services can be represented for use with web services and stored in a reusable way. In Section 5, the utilisation of support services is illustrated with four implementation examples of support services in combination with generalisation operators that use the output of these support services. The examples have been chosen so as to form a representative subset of the main types of support services found in the proposed taxonomy. Sections 6 and 7 end the paper with a discussion and conclusions, respectively.

2. Generalisation Web Services

This section briefly reviews the main elements of previous work (Burghardt et al. 2005, Edwardes et al., 2007) to provide the basis for the discussion of the following sections.

2.1 Interoperability through Web Services

There are several advantages of using a Service Oriented Architecture (SOA) such as Web Services for generalisation. In an SOA environment, resources on a network are made available as independent services. These services interoperate according to a formal specification and can be accessed without knowledge of their underlying platform implementation (Channabasavaiah et al., 2003). The platform independence makes the development independent from the operating system and the hardware used. Furthermore, services can be integrated into any software platform, such as web browsers, GIS, or map production software (Regnauld, 2006). It is even possible to write specific algorithms for special computer architectures such as clusters, grids, or other parallel processing systems and to offer such services to the subscribers (Burghardt and Neun, 2006).

Services extend the traditional approach of providing a software library with a well defined interface. Software libraries can often only be used in one particular programming language. Services however allow the interoperable use with different languages on different platforms. Thus services are more flexible, exchangeable and also comparable. Valuable functionalities from different platforms can be combined. Finally, (web) services can be accessed either over the internet or locally.

Web Services are not only useful for accessing data over the web but also for accessing processing functionalities on a remote server. The Open Geospatial Consortium Web Processing Services (OGC, 2005) are a first step towards web services for the execution of spatial algorithms, including those for generalisation. So far, however, there have been no specifications for the ‘Feature Generalisation Service’ mentioned in OGC (2002). The starting point for such a

Generalisation Service should be some suitable small services (ICA, 2004) without having to deal with the harmonization of data types and structures (Lehto and Sarjakoski, 2004; Illert and Afflerbach, 2004). For interoperable services for the visualisation of spatial information (Fitzke et al., 2004) this has already been demonstrated, yet not with generalisation.

2.2 Types of Generalisation Services

The overall generalisation process involves both rather simple independent generalisation operations, which are applied only to individual map features, as well as highly context-dependent operations, which require comprehensive control over the generalisation workflow. Hence, Burghardt et al. (2005) argued that three types of generalisation services must be offered in order to enable comprehensive web-based generalisation. These three categories are shown in Figure 1 and briefly discussed below.

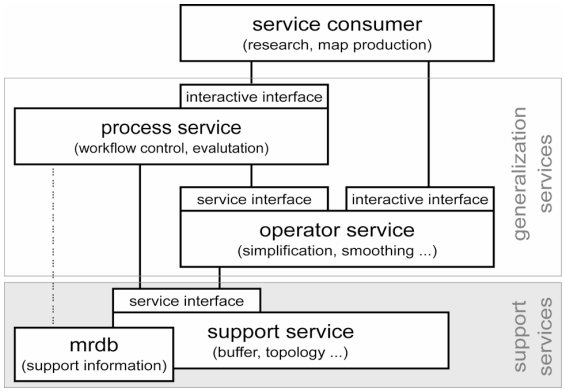


Fig. 1: Categories of generalisation services, the highlighted support services form the focus of this paper

2.2.1 Generalisation Support Services

Support Services can be seen as a provider of additional (enriching) cartographic information in support of the automated generalisation process. Examples are services for buffering or for creating a topological data structure, a skeleton or a constrained Delaunay triangulation (details see § 3). Support services take raw cartographic data with a simple structure as input and deliver either a simple structure but with additional enriching information (e.g. a priority ordering) or a more complex data structure with object relations as output (e.g. a graph data structure representing adjacency relationships between map features). Thus a main goal of such services is to make structural information explicit, representing common structural properties such as alignments, neighbourhood or proximity relations, which can be usefully exploited by generalisation operations. Support services can also be used for calculating measures and constraints. Sometimes the establishment of such supporting information is very expensive in terms of time and memory so that optionally their persistent storage in a special multi-resolution database (MRDB) can be useful e.g. for real-time generalisation. Support services together with multi-resolution databases are the fundamental service category, which offers its functionalities to all other service types (see Fig. 1). The representation and analysis of relations between map features, such as topological or metrical relations, are a fundamental part of GIScience (Worboys and Duckham, 2004). Therefore it is important to note that support services such as the ones discussed in § 3 could be equally useful in a more general, non-generalisation environment, particularly for the purposes of spatial analysis and decision support.

2.2.2 Generalisation Operator Services

Operator Services deliver the functionality of standalone generalisation operators such as the ones defined by McMaster and Shea (1992). Examples are services for simplification, smoothing, aggregation, amalgamation, merging, collapse, refinement, exaggeration, enhancement and

displacement of cartographic features. These generalisation operator services can be further subdivided for point, line and area features and specialised depending on feature classes. Operator services themselves can use the functionalities of support services. For instance, a feature displacement service may make use of a triangulation data structure delivered by a triangulation support service. Generalisation operator services form a service-oriented toolbox for realising generalisation processes. In the current version of the WebGen prototype implementation more than 15 operator services are available and are continually extended.

2.2.3 Generalisation Process Services

Process Services use services from the lower categories for the control and guidance of generalisation operators (Fig 1). These include services for automated generalisation workflow control, as well as services for the evaluation of generalisation results. Automated control of the generalisation process presently receives ample attention as a research topic. Approaches for process control currently favoured in the map generalisation community include optimisation techniques such as simulated annealing (Ware et al., 2003) and energy-minimization (Bader et al., 2005) as well as agent-based methods (Barrault et al. 2001). For a review of such methods see Harrie and Weibel (2007); for a review of early work in this area see Buttenfield and McMaster (1991).

2.3 Generalisation Service Architectures

Generalisation services are delivering generalised data to the requesting client. But where does the original un-generalised data originate from which is then generalised by the service? In general two main scenarios are possible.

The first scenario is that the data comes from a geographic database, usually via a Web Feature Server (WFS). Thus this approach is merely an extension to a database. This ***middleware generalisation service*** delivers pre-generalised data. Such an approach is described by Sarjakoski et al. (2005) using a modified WFS. A possible use is an adaptive zoom for Web Map Services as it is currently the domain of multi-resolution databases (MRDB). This service would require a fully automated, real-time execution of the generalisation process. Another possible use are generalisation support services (the focus of this paper) which pre-process cartographic data e.g. from a WFS so that they can then be used by more complex generalisation algorithms.

In contrast, the second scenario foresees that users can generalise their own data, by sending them to the service. Such a ***generalisation toolbox service*** offers its processing capabilities including algorithm logics and computational power to the user. The ability to provide specialised or novel algorithms in a platform independent way allows the evaluation and integration of generalisation functionalities from different sources without forcing the users to adapt their systems to any specific needs. In an open toolbox service model everybody can present his/her own generalisation services. Through the Internet and the use of platform independent technologies such services can reside on servers all over the world. For discovering these services a Registry indicates which services are available, where they are located and what algorithms they offer.

Possible usage scenarios include research as well as map production. The map generalisation research community has an interest into generalisation services, driven by the desire to develop a common open research platform that would allow testing and sharing of generalisation algorithms (Edwardes et al., 2003). For the distributed and collaborative development of generalisation techniques the service model can help the coupling of algorithms and data structures on different platforms. The evaluation of algorithms etc. is facilitated. Generalisation toolbox services can support research cooperation by sharing of techniques within the cartographic research community (Regnauld, 2006), for example new algorithms, enriching data structures or measures.

As the service's logics reside only on the server of the owner even copyright protected methods can be offered, evaluated and incorporated without having to give executables or even code away. Hence, in map production services can help the coupling of heterogeneous production platforms in order to generate a seamless generalisation workflow. Further scenarios would even include offering specialised services on a per-use basis with a fee to occasional users of generalisation functionality.

2.4 Generalisation Toolbox Service Platform

The prototype of a generalisation toolbox service platform is the WebGen framework. Initial implementation details were reported in Neun and Burghardt (2005). The WebGen platform consists of client and server components. The client plug-in offers the configuration and selection of the desired service. The data to be processed by the service together with the parameters for the algorithm are encoded and sent to the service. The result, returned by the service, is then decoded and presented in the client. Currently client plug-ins for the JUMP Unified Mapping Platform (2006), a client as a browser based AJAX web page (Asynchronous JavaScript and XML) and a prototype plug-in for the Clarity generalisation software by 1Spatial (2007) are available. A client with similar functionalities could also be developed for other desktop GIS having an API for adding functionalities.

The first implementation of a generalisation server for the WebGen framework uses JAVA. Services written in JAVA can be offered directly. Algorithms written in other programming languages or algorithms that are available as executable programs can also be included with full functionality using file-based data transfer. Every service can also make calls to other services on the same or on another generalisation server. A simple example is that an algorithm, needing a buffer around a point, can send the point geometry to the buffer service, receiving back a buffer geometry.

Initially rather simple services (e.g. Douglas-Peucker line simplification, building simplification) with no consideration of spatial context were implemented in order to demonstrate the feasibility of the service-based approach. In this paper more advanced service types and their implementation will be described, with a focus on generalisation support services that form the foundation of structure recognition in map generalisation.

3. Taxonomy of Support Services

The execution of generalisation operators or algorithms depends on the input they receive. Important elements include algorithm parameters, the character of the map features to generalise, and also mutual influences between map features, such as roads exerting a push on nearby houses in map feature displacement. Thus the knowledge of the spatial context is a very important factor for generalisation (Mustière and Moulin, 2002). Most of this knowledge about relationships is only implicitly contained in off-the-shelf cartographic data sets. Generalisation algorithms usually have to create this knowledge about the spatial context in the form of auxiliary data structures during their execution. Examples range from the creation of a simple buffer to a topological data structure, a skeleton or a constrained Delaunay triangulation.

In this paper, we are assuming that we are dealing with vector cartographic data as well as vector data structures and algorithms. Raster cartographic data and algorithms are rarely used in map generalisation (e.g. with raster land-cover data). Hence we restrict our following taxonomy to the vector domain.

As mentioned above support services enrich map data with additional information such as cartometric measures, contextual knowledge such as relations between map features or also additional meta-data. Many of the measures and contextual data structures that form elements of our taxonomy of generalisation support services are available in some form in commercial GIS platforms. Hence, one might think that starting off from commercial GIS might be a suitable approach to develop appropriate supporting functions. However, there are two main reasons that speak against that approach. First, commercial GIS developers tend to favour functionality that satisfies the interests of a broad customer population, while map generalisation as well as other GIS fields such as spatial analysis or spatial queries often requires adapted and specialised forms of supporting data structures, as will be shown below. Hence, one of the intentions of proposing our taxonomy is also to show the breadth of support measures and data structures needed in this particular field and show their utility, in order to stimulate interest in developing these functions in GIS packages. Second, GIS platforms often have a rather monolithic architecture. Using web services the enriching data can be made available in a platform-independent way which allows the combination and evaluation of different services using a common interface. The developer can use

support services as components for his/her system, relying on the defined interface of each service, without the need to rewrite code. It is potentially also possible to use multiple support services, even from different providers. If the service and the service consumer reside on the same computer, the services can also be accessed locally without network overhead and with the reliability that the service is always available. Support services can be used for data pre-processing with expensive calculations as well as for real-time generalisation.

The taxonomy of generalisation support services presented below tries to identify commonalities and differences between services. Different levels of complexity exist for support services in terms of their output data. Simple support services just do create supporting entities like geometries or attributes. More complex services create information about the relations between map features or between their properties. Thus categories of support services can be distinguished by the type of the supporting data they offer, and thus by the output they deliver to the service consumer (Fig. 2). Using the output complexity of a support service for the classification identifies an important commonality/difference between the different support services which can be used as a framework with different levels of complexity. The output defines also the complexity of the interface which is needed to access such a support service. In order to be able to build a common 'library' of support services for generalisation such a framework can help the distributed development process with the involvement of many different parties in the generalisation community.

There are two main types of support services. First, there are those services that equip entities (i.e. map features) with new or modified geometries or attributes (see § 3.1 for details and examples). Second, there is the large family of Relations (see § 3.2). Relations can exist between objects but also between properties like object states. They range from (directed or undirected) graphs, such as transport graphs and triangulations, to hierarchies which can be expressed by trees. All graphs and trees can also be represented as a matrix.

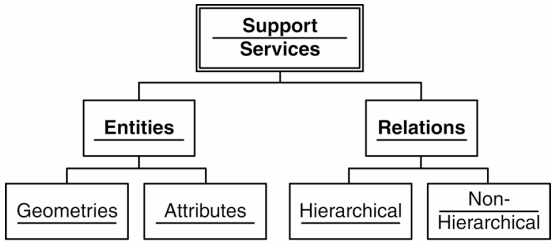


Fig. 2: Support service types (categorised by service output complexity)

3.1 Entity Support Services

The most obvious output of a support service are entities with supporting attributes or geometries in the form of the simple features (OGC, 1999). These services are just enriching features with additional attributes or are creating new support geometries. Functions to read and write these supporting data structures are included in most GIS software packages. Consequently, the discussion below will be brief.

3.1.1 Creating Geometries

The output of such a support service is just the derived geometries. OGC-style simple features can easily express them. Thus, no extra data format is needed to get results from the service.

A simple example of a geometry creating support service is the creation of the buffer polygon from an input geometry, which could then be used by a feature selection service. Taking, for instance, a road and a set of houses as input a selection service may return the houses contained in a certain buffer around the road. The output of both services is just simple features. A further example of supporting geometries includes the computation of alignment lines, chaining together a group of map features such as buildings (Christophe and Ruas, 2002). The creation of inflection points or identification of local extremes for line generalisation (Plazanet et al., 1995) serves as a final example of creating supporting geometries. It delivers a series of critical points which can

then be used, among others, for line segmentation (partitioning) or the creation of trend lines, approximating a line by connecting the inflection points (Fig. 3).

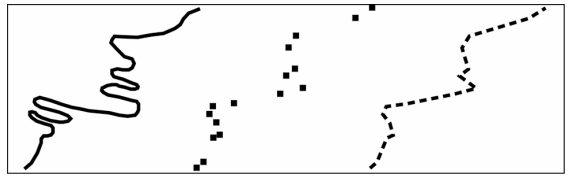


Fig. 3: Original line; inflection points; trend line generation

3.1.2 Generating Attributes

These services take map features as input and return the features with modified or new attributes. In essence, most of these services are performing an analysis of the shape and structure of map features, also termed cartometric analysis (McMaster and Shea, 1992). An example of such a function is the calculation of the sinuosity of a line (Plazanet et al., 1995), which is stored as an attribute of the line feature. Plenty of other measures can be calculated, such as area or shape index of a polygon, shortest distances to nearest neighbours, and many more. Often, these measures can be used in a comparative way to establish priority orderings among map features (e.g. small polygons may be defined as insignificant and therefore omitted). The calculation of measures or constraints (Burghardt and Neun, 2006) can be used in an automated generalisation process for choosing appropriate operators or for evaluating algorithm results. Thus a support service can calculate for example the cost of the violation of a minimum building size constraint and attach this value as an attribute to the buildings of a map. Such a severity value can be used to trigger the appropriate generalisation operator and the parameters for every feature individually. If for example the minimum building size constraint is violated the building feature must be enlarged until the constraint is satisfied.

3.2 Relation Support Services

Spatial, structural and semantical relationships are essential ingredients for many complex generalisation operators (Mustière and Moulin, 2002; Regnauld, 2005). The corresponding relation support services have a more complex and context dependent output than the entity support services. Despite the fact that the value of such relational structures is well known they are unfortunately only sparsely available in commercial GIS and if available they are often not generic and rich enough to be exploitable for the purposes of map generalisation.

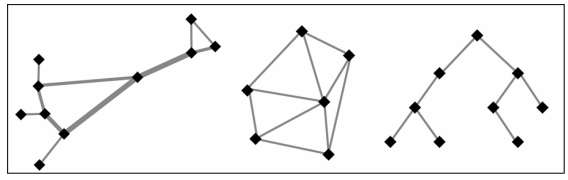


Fig. 4: Examples of graphs: Weighted graph, triangulation and tree

In general relations between cartographic features can be modelled as graph-like structures (Fig. 4). Examples are topological data structures (e.g. polygonal maps), transport and neighbourhood graphs, triangulations, or surface networks. As a more specific case, hierarchies can be expressed by trees being a special form of a graph with no cycles, rooted in a single node. The modelling and implementation of such graphs will be shown in §4.3.

In the next two sections hierarchical relations and non-hierarchical relations, respectively, are reviewed. For the sake of brevity we exclude relations for 3-D objects (e.g. terrain surfaces), though obviously both hierarchical and non-hierarchical relations can also be found in more than two dimensions. Examples of hierarchical relations for surfaces include hierarchical TINs (De

Floriani and Puppo, 1995); the prime example of non-hierarchical relations in 3-D are surface networks (Pfaltz, 1976; Rana, 2004).

3.2.1 Hierarchical Relations

Hierarchical Relations can exist between cartographic features. Hierarchy creating criteria may be any property of the features involved such as their relative positions or an attribute such as a class scheme. Hierarchical relations can be expressed by trees. In some cases the hierarchy creating criterion can be a semantical property as with similarity trees or dendrograms generated by classification or clustering algorithms. In other cases a hierarchical relation represents a structural property as with network orderings, complex features, or reactive trees. Hierarchical relations can also be seen as a more specialized type of relation than the non-hierarchical relations.

Similarity trees or dendrograms: For the aggregation of geometries or the translation of features from one feature schema to another, often a reclassification of the feature categories is needed. A reclassification needs some sort of rule set on how to assign new categories to the input features. This rule set in many cases represents a strict hierarchy which assigns a new category to every input category (e.g. 'deciduous forest' and 'coniferous forest' are both reclassified to 'forest'). This hierarchy can be defined by the user of the system or generated automatically, for instance by a statistical evaluation of the properties of the input categories. A reclassification service would request a similarity tree from the support service and then classify the map features accordingly (see example in § 5.3). Thus a similarity tree expresses the semantic similarity or adjacency of the feature categories, which can then be used for reclassifications or aggregations (van Smaalen, 2003). A special case is encountered if multiple output categories are possible for a given input category. However, in such a case these dependencies are no longer strictly hierarchical and a weighted decision graph or a transition matrix can be used instead (see 3.2.2).

Hierarchical network ordering: A well-known example for a hierarchy is the Horton-Strahler order of hydrological networks (Horton, 1945; Strahler, 1952). This is a very obvious and intuitive example to represent a river network as a tree structure from the outlet to the source links (except for a few exceptions such as braided streams). This ordering does not have to be returned by the support service as an explicit tree data structure but it could just be added to the attribute table of the river network.

Reactive data structures and Levels of Detail: For the efficient storage and access of line simplifications or polygon aggregations in MRDBs tree structures provide a useful hierarchy for the features involved. Van Oosterom (1994) proposed tree structures for on-the-fly generalisation including the Reactive Tree for storing and retrieving map objects at multiple detail levels and the BLG tree for line approximation by the Douglas-Peucker algorithm in a multi-resolution environment. Van Oosterom and Schenkelaars (1995) further describe the GAP Tree, useful for polygon aggregation in polygonal subdivisions. For the generation of multi-resolution databases (MRDB) out of different datasets the *matching* of the features on the different Levels of Detail (LoD) is indispensable. The links between the features on the larger scale with the matched features on the reduced scale are mostly of nature 1:0, 1:1 and 1:n as shown by Timpf (1998). 1:1 and 1:n relations are partonomic relations (Bobzien et al. 2006) and can be expressed by a simple tree. These ‘vertical’ links (Neun et al., 2004) between the map features on different LoDs help the automated propagation of changes while updating the MRDB. However, n:m relations as they exist, for instance, in typification operations (e.g. 5 buildings are typified to 3 buildings) are more complex to model and require a DAG (directed acyclic graph) as they cannot be modelled by a tree.

Complex features and partitioning: Map features often form meaningful groups, that is, complex map features consisting of simple features. Examples include a cluster of buildings that form a hamlet; an alignment of similar buildings along a road; a group of buildings and surrounding streets forming a city block; or several fields, ponds, trails, etc. forming a park. Since complex features represent groups they are the simplest and also most general case of hierarchical (partonomic) relations. Often, however, complex features such as the above examples are not stored explicitly in cartographic databases. Thus, they have to be created either interactively by user definition or

automatically by cartographic pattern recognition (e.g., clustering procedures). The knowledge about these partonomic relations informs the selection of appropriate generalisation methods and parameters in order to preserve the original structure of the map. Groups of map features can also be derived through spatial partitioning of map data into regular or irregular tiles. For instance, settlement generalisation is facilitated by partitioning the settlement into city blocks formed by the street network (Ruas 2000). Additionally, such partitions may also be exploited to increase the speed of complex generalisation operations by reducing the amount of features that have to be generalized at once. Also, the distributed processing of a partitioned dataset on multiprocessor computer architectures or even on clusters is possible (Burghardt and Neun, 2006).

3.2.2 Non-Hierarchical Relations

Common representations of non-hierarchical relations are graph data structures and matrices (see 4.3). The graph structures may contain cycles, are possibly weighted, and/or directed. The support services in this category describe the spatial and semantical relations between cartographic features in diverse ways.

Minimum spanning trees (MST): Although it is named a tree the MST does not express a hierarchical relationship between the features it connects. The MST can be used for a variety of purposes in generalisation. It has been used, for instance, for the selection of candidate roads in automated road matching for multi-resolution databases (Lüscher, 2005). Bader et al. (2005) use a *ductile truss* for managing building proximity relations in the building displacement process. This elastic truss connects the building centroids, adding further edges to a MST and forming cycles until every building is connected to at least two neighbours. Roads or railway networks form graphs as well. Such transport graphs can be used to generalise a road network connecting a set of cities by selecting roads in variants of the minimum spanning tree (Mackaness and Beard, 1993; Thomson and Richardson, 1995). Hence the connectivity of the transport network is assured.

Topology graphs: For expressing direct adjacencies between map features topological data structures are used. A topological data structure is an extended planar graph and uses nodes, edges and faces to represent the topological relations of map features. Thereby the space is subdivided completely by the nodes and edges of the map features. The use of such a topology graph in an algorithm ensures data integrity, shared boundaries and connected networks. For example, in generalisation topological rules (embedding, planar enforcement, etc.) can be used to prevent holes which are created due to a geometry type change from a polygon to a line or point (Bobzien and Morgenstern, 2002). A data format for interoperable storage and exchange of topological data structures is available through GML 3 (OGC, 2004).

Neighbourhood or proximity graphs: For expressing the relations of a feature with its neighbourhood in a more general way than with the pure topology structures such as a nearest neighbourhood graph or a relative neighbourhood graph can be used. Thus neighbourhood graphs can also express relations where the generalisation of a feature influences its surrounding features though they are not directly adjacent. Anders (2004) used neighbourhood graphs for the interpretation of spatial data, data analysis and data enrichment of disjoint objects (e.g. buildings). For the establishment of neighbourhood graphs very often triangulations between the features are used. Regnauld (2005) proposes a triangulation and a proximity graph that extends the triangulation of feature centroids with the true distance between feature geometries (an application is shown in Fig. 12 below). This proximity graph can be used to derive clusters of close features but still maintains a relation between all triangulated features.

Triangulations and related structures: The *Delaunay triangulation* of a point (vertex) set (Fig. 5) is a collection of triangles satisfying the property that no other vertices are within each triangle's circum-circle. The constrained and the conforming Delaunay triangulations are adaptations of the Delaunay triangulation which can be used to triangulate over polygonal objects by incorporating the polygon edges as predetermined or constrained triangle edges.

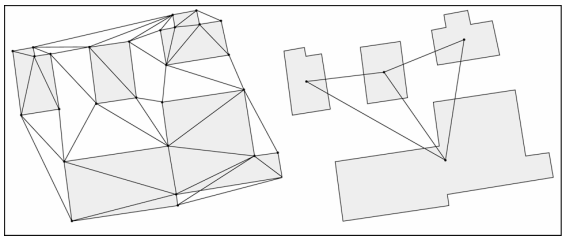


Fig. 5: Delaunay triangulation (non-constrained) of polygon vertices (left) and polygon centroids (right)

Jones et al. (1995) use the constrained Delaunay triangulation in their simplical data structure (SDS) for implementing exaggeration, collapse, amalgamation, reduction and displacement algorithms. Ruas (1998) uses a Delaunay triangulation of building centroids (see example Fig. 5 right) to represent proximity relations for managing the building displacement process. Thus triangulations can be used for expressing neighbourhood and proximity relations as shown by Burghardt and Cecconi (2007) for the typification of buildings.

The *Voronoi diagram* is the geometric dual to the Delaunay triangulation. The constituent Voronoi polygons are also known as Thiessen polygons defining the border of the space which is closer to the contained object (e.g. a point) than to any other object. Thus, the result is a complete tessellation of the space between the objects. Chithambaram and Beard (1991) use these properties for creating the *skeleton* of a polygon. The skeleton is of interest in generalisation as it represents the 'interior structure' of polygons. Hence, the skeleton – and more specifically, the straight line skeleton – has been used by several authors for purposes such as polygon aggregation in polygonal subdivisions (Bader and Weibel, 1997), collapse and partial collapse of polygons to lines (Haunert and Sester, 2004), and road centre line generation (Petzold et al., 2005).

Decision graphs: Strictly partonomic relations can be expressed by hierarchical trees. Partonomies with multiple associations, however, cannot be represented in a tree because of the possibility of cycles in the structure. As shown in Fig. 6 an alley could, depending on its size, become part of a city block or part of the road network. A weighted decision graph (or a weighted adjacency matrix) can express such relationships. Such a graph can also be represented as a directed acyclic graph (DAG) which has, if needed, weighted edges.

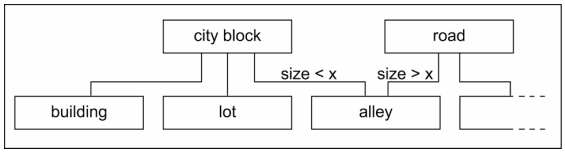


Fig. 6: Partonomy with multiple associations

The modelling of such context-dependent relationships often involves semantical as well as geometrical and structural properties. Context-dependent decision graphs have been used as a rule-base in model generalisation for transforming feature schemata (Bobzien, 1999), and also for the reclassification of features, preceding aggregation or amalgamation operations.

3.3 Summary of Support Services for Vector Data

Above, we have proposed and populated a taxonomy of support services to aid the generalisation of vector cartographic data. These generalisation support services are summarised in Table 1. Selected implementation examples of representative services are shown later in § 5.

Entity Support Services		**Relation Support Services**	
Creating Geometries	**Generating Attributes**	**Hierarchical Relations**	**Non-Hierarchical Relations**
- buffering - partitioning - creation of alignment lines - creation of inflection points	- calculation of line sinuosity - area - shape index - various cartometric measures - classification (§ 5.4)	- similarity tree or dendrogram (§ 5.3) - LoD links in MRDBs - network ordering - reactive data structure - complex features	- minimum spanning tree (MST) - transport graph - topology graph - neighbourhood graph (§ 5.1, § 5.2) - triangulations and related structures (§ 5.1) - decision graph (§ 5.3) - weighted matrices (§ 5.4)

Table 1: Support service types

While we argue that our taxonomy is comprehensive, it is by no means meant to be populated exhaustively. Besides providing an overview of the kinds of functionalities needed in support of generalisation operations, the main purpose of building this taxonomy is to allow to devise an appropriate set of data structures for support web services. Therefore, we tried to find commonalities of the data structures needed for the representation of the entities and relations involved. The output of the four support service types differs in terms of complexity and availability for representation in a data structure, for storage in a file format and for transfer over networks. Data structures for storing, exchanging and utilizing the supporting knowledge offered by support services are discussed in the next Section.

4. Storing, Exchanging and Utilizing Data from Support Services

Generalisation support services should deliver enriching data structures and information which are otherwise expensive to calculate or difficult to implement on the target platform. Thus they should facilitate the development of new algorithms by providing output from complex auxiliary algorithms, such as complex measures, auxiliary geometries or map feature relations, with an easy interface and in a preferably simple format. The aim is to make support services available to developers of service-based generalisation architectures in such a way that they can use these in conjunction with generalisation operator services without having to know in detail how the auxiliary data are generated. For support services that are complex and expensive to create a persistent data format is desirable in order to allow reuse of the supporting data structures in a workflow of generalisation operators in the web services environment.

4.1 Deploying Entity Support Services

The two classes of *Entity Support Services* are well covered by existing standards. Supporting entities are either geometries or additional attributes for map features. For the representation of geometries the Simple Features (OGC, 1999) are well known and widely used. Formats such as Shapefiles (ESRI, 1998) or GML (OGC, 2004) can be used for map features with attributes. Owing to these widely used formats the supporting knowledge in these entities can be exploited with every standard GIS.

4.2 Deploying Hierarchical Relation Support Services

For the representation of hierarchical tree-like relations between map features or between their properties standard geometries and attributes are usually not sufficient. Therefore other formats for representing and exchanging hierarchical structures are needed. XML is a very flexible format and has an implicit hierarchical structure. Thus it can be used for representing hierarchies directly. Elements of an XML data structure can enclose other elements. Hence they are creating a nested structure, which represents a strict hierarchy. Such an XML representation can also be used in an object-oriented manner in the main memory of a computer. In doing so almost all XML parsers for object-oriented programming languages are offering a tree-like data structure for querying the content of an XML document. Thus XML offers a format for storage and network transfer as simple text and, when using an XML parser, an object-oriented format that can be queried, modified and kept in main memory.

A similarity tree can be modelled using the hierarchical nested structure of XML. Listing 1 shows the simple code example of different forest types which belong all to the parent class 'Forest'.

```
<webgen:Hierarchy>
    <webgen:Node value="forest”>
        <webgen:Node value="deciduous forest” />
        <webgen:Node value="coniferous forest” />
    </webgen: Node >
</webgen:Hierarchy>
```

Listing 1: Simple XML similarity tree for reclassification

The relations between the different classes can easily be queried by requesting the parent or child nodes of the XML tree. Thus when querying the XML document with a standard XML parser the node "deciduous forest" has a pointer to its parent node "forest". An application example of such a hierarchy is given later in § 5.3.

4.3 Deploying Non-Hierarchical Relation Support Services

The available output formats for *non-hierarchical* relation support services are much more heterogeneous. Non-hierarchical formats require other, more complex (and diverse) data formats. In GML 3 (OGC, 2004) a format for topology graphs is available. Ways for representing other non-hierarchical relations such as triangulations using graphs or matrices are discussed in this section.

Non-hierarchical relations can be represented by graph-like structures. In a basic object-oriented representation (Fig. 7) a graph consists of a list of nodes and a list of edges. Each node contains only a list of the incident edges. The edges have two variables, which point to the two end-nodes of the edge.

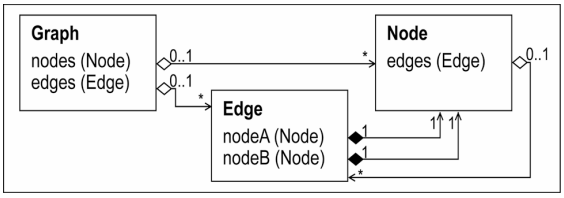


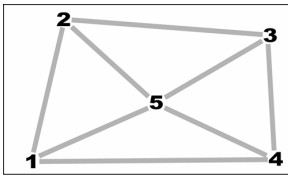
Fig. 7: Basic object-oriented graph representation

For its use as a generalisation support structure for different sorts of non-hierarchical relations such a basic graph can be easily extended and specialised through class inheritance. In graphs that represent cartographic features, where the nodes correspond to nodes or vertices of cartographic features, the nodes have coordinates and/or contain a link to the corresponding cartographic feature. For planar graphs like topology graphs or triangulations the basic graph model can be

extended with faces or triangles. An example of such an extension for Delaunay triangles is shown by Regnauld (2005).

This in-core representation is optimized for the querying and modification of the graph but it can only be used in the main memory of a computer. Thus a transfer data format is needed that is optimised for efficient storage, transfer over a network and parsing on the receiving system.

Non-hierarchical relations can also be represented as matrices relating every feature to all others. A matrix is basically the dual of a graph; every graph can be expressed by a matrix and vice versa (see Fig. 8). But using a matrix for representing a graph with many nodes but only few edges per node is very inefficient in terms of data size. This is because in a matrix every possible edge, also the ones that do not exist, is represented. Thus for a complete graph or another structure where every feature is related to all others also a matrix representation is an effective in-core representation.



	1	2	3	4	5
1	0	1	0	1	1
2	1	0	1	0	1
3	0	1	0	1	1
4	1	0	1	0	1
5	1	1	1	1	0

Fig. 8: Simple graph and matrix representation

More efficient data formats or languages for storing and exchanging graphs are for example very simple text files containing node and face (i.e. triangle) lists (Shewchuk, 1996) or the XML based Graph eXchange Language GXL (Holt et al., 2000). GXL is very expressive and tries to represent a maximum of different graph types. For representing specific graphs also own tailored list formats using e.g. XML can be created by representing all edges as tuple of the identifiers of their two end-nodes. All these serialised data formats are based on a list representation such as an adjacency list for simple graphs or a triangle list with associated nodes and possibly their adjacent triangles. The advantage of these data formats is that they are compact, they do not contain much redundancy and they can easily be saved in a file or be sent over a network using standard protocols.

However, using these list representations as an in-core data structure for graph representation is not very practical and efficient. An object-oriented data structure is much easier to query, modify and extend. Thus when deploying complete graph structures with web services using XML a conversion between the in-core representation and the transfer format has to be carried out and vice versa.

Also matrices can be represented using XML structures. Usually in this case table-like XML representations are more compact and better to parse than the edge lists used for representing graphs. Such a matrix representation in XML can even be exploited directly by traversing the XML structure without having to parse the complete structure.

In the WebGen research platform XML and HTTP are used for the data exchange between the different services and the client, making the usage of an XML based graph data format straightforward. As there exists within GIS no specialised standard format for representing graphs, a data format which can easily be created and read in different programming environments or GIS platforms is desirable. Thus, parsers for the different generalisation support data structures are needed. In the case of an object-oriented representation of the graph on the computer the data structure is converted to an XML structure, transferred over a network and parsed on the receiving system into its own, internal data structure, which may not necessarily be the same as the transfer format. In the WebGen framework we are using an XML representation for the transfer of graphs which is based on a redundancy free adjacency list. This XML representation can be parsed efficiently into an object-oriented model and vice versa. As there is no redundancy the data volume depends directly on the number of nodes and edges. This ensures a fast network transfer of the graph. For transferring a matrix of decision borders (see § 5.4) a table-like XML structure was used and proved to perform well.

4.4 Stateful Handling and Deployment of Support Services

If multiple generalisation services are used in a workflow when all the data has to be passed from one service to another and if a particular service is subsequently used with the same data but only slightly changed parameters, this will cause massive transfers of data over the network. Thus, instead of using normal stateless web services a stateful approach can be helpful. Stateless services have no memory to remember preceding calls and their parameters and data. Conversely, a stateful service creates a session for every user where settings, parameters and data can be saved and retrieved when the service is called multiple times.
The advantage of *stateless services* is their simplicity. The service has only one input source for parameters and data. The data which is transmitted may consist of simple feature data but also of any other data structure which has for instance been recalculated by another support service. However, the amount of data which has to be transmitted upon every request can be quite high. Thus the stateless services are more suited for testing environments where the datasets are rather small. When calling a support service the result may, for example, be a complex graph structure in an XML format as described in § 4.3. Such a representation may be used by other services. To query and extract information from such an XML format is usually simple and can be handled on the client. However, if the graph has to be modified also the logics to insert, delete and change nodes must be available on the clients.

A *stateful support service* remembers old data or parameters and can access and use them upon being called again. Such a service creates a session for every calling client. Upon a new call old settings and data structures can be requested and reused. For example a triangulation service can retain the triangulated structure persistently and the triangulation may be queried or modified again later. Thus the calling client calls the support service once with the input data and parameters. After this initialisation step the generated data structure is kept persistently on the server. Later on the calling client may query the data structure, for example, by retrieving the whole graph or by only demanding a single edge and its neighbours. The calling client may then also send requests to insert, delete or modify a node. The functionalities of such a stateful support service fundamentally depend on the functionalities of the implemented support data structure.

Alternatively the parameters and data may also be kept persistently in some sort of more generic stateful service where other services residing on different servers can access it. When dealing with large datasets and when a service chain is involved with a workflow of services that are executed sequentially a stateful environment avoids network traffic and removes load from the client. In this scenario a client may be either the calling user program or another service which would then be a workflow service which is triggering the single services on the service chain. Such a stateful service could be an extended, transactional WFS which would then store control parameters, map data plus supporting data structures. All algorithms in a particular workflow could then access this store and retrieve the data they need for their execution, subsequently updating the store.

In Fig. 9 an example sequence for the use of a stateful transaction server for spatial web services is shown. Here the session is initialised once and the data (map features and other parameters) which will be treated in the following workflow are uploaded. After that the data can be accessed and modified by other services. Hence the transaction server is basically an extended network-enabled data model for storing feature data and supporting data structures. The advantage is that during a generalisation process not all the data have to be passed from one generalisation service to another. With this approach the individual services are only retrieving the data they need, storing their results back on the transaction server.

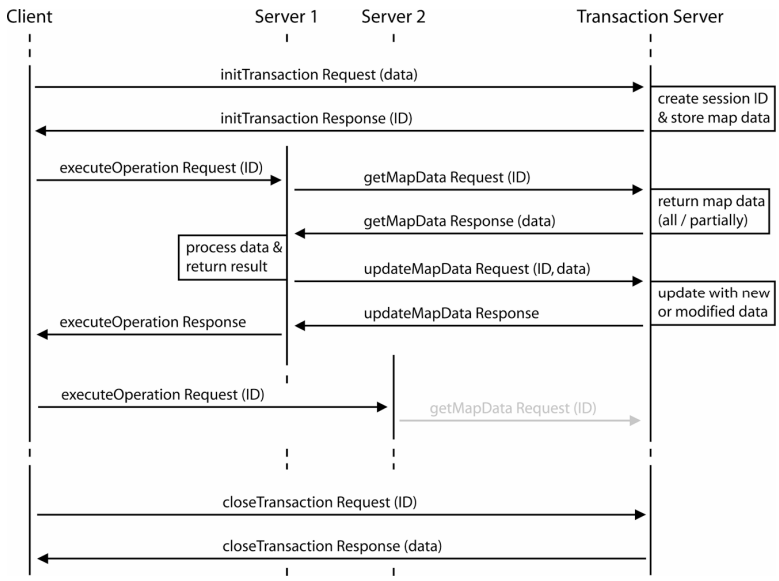


Fig. 9: Transaction server architecture for support data structures. Requests of the second server are set in gray font or omitted.

In collaboration with other stateful support services this transaction server can also act as an authority to manage the data seamlessly by providing unique identifiers and matching tools to merge enriched data from different sources.

Some of the implementation examples presented in following section are implemented as stateful support services. These include a triangulation, a proximity graph, alignment groups, and a matrix that are persistently modelled in the service and can thus be queried and modified.

5. Implementation Examples of Support Services

In this section selected examples of some of the generalisation support services which are implemented in the WebGen framework are explained. These examples demonstrate possible usage scenarios and benefits of using support services. They have been selected so as to provide a representative sample of the functionalities and data structures employed by the proposed taxonomy of generalisation support services.

- As an example for non-hierarchical relations a stateful service for triangulations and proximity relations is shown. This service is subsequently used by a building typification and a building displacement algorithm, thus demonstrating the collaboration of support services and operator services (§ 5.1).
- As an example for structural relations expressed by non-hierarchical relation a service for the detection of building alignments is shown. This service can be queried for alignment relations and can return aligned buildings as groups or complex features (§ 5.2).
- As an example for spatial and semantical context represented hierarchically or as a decision graph a service for the reclassification of features and their schemata is described. This service creates and changes attributes and can be used for aggregating land cover data. Additionally a conditional (non-hierarchical) decision graph for these schema transformations is presented (§ 5.3).
- The final example presents a service for supervised classification of buildings into urban structure types. The building classifier is implemented as an entity support service returning attributes assigned to each building feature. As it is a supervised classifier, this service in turn calls a training service returning a matrix representing relations between building shape measures (§ 5.4).

5.1 Proximity Relations for Building Generalisation

In this section we present a stateful support service serving triangulations and further proximity relations. The proximity graph extends a Delaunay triangulation of the centroids of map features (Fig. 10) by additional measures for the 'real' distance between two geometries (Regnauld, 2005). Thus for every edge in the triangulation linking two map features an additional edge in the proximity graph exists. In Figure 12 a proximity graph between buildings as well as buildings and roads is shown. In order to make the combined proximity and triangulation graphs available to other services they are implemented as a stateful support service representing the graph persistently and offering functionalities to query and modify the graph structure. This support service will be used subsequently by two operator services, a typification and a displacement algorithm for buildings. Hence, these two operators are first briefly explained.

The typification algorithm adopts the idea of Burghardt and Cecconi (2007) and uses the Delaunay triangulation of the building centroids supplied by the support service (Fig. 10) for deriving which buildings are closest to each other. In an iterative process this shortest edge is then collapsed and the buildings are replaced by a placeholder. After that collapse the triangulation is updated and the new shortest edge is derived. This process continues until the desired number of buildings has been replaced. The edges of the triangulation are weighted according to the size of the two buildings they connect. Thus an edge between two small buildings is shorter (i.e. has a lower weight) than an edge between two large buildings and the two smaller buildings are replaced first by a placeholder. For generating the placeholder, the building with the larger area of the two collapsed buildings is chosen, enlarged in order to meet the same black-to-white ratio and positioned at the centroid of the two buildings (Fig. 11). This method works well in semi-dense, suburban and rural areas. In other areas an amalgamation algorithm could be used instead, for example.

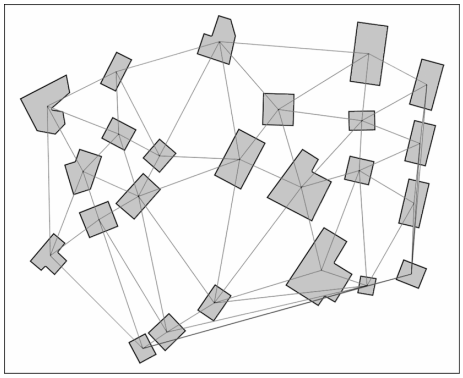


Fig. 10: Triangulation of building centroids (reproduced by permission of swisstopo, BA071153)

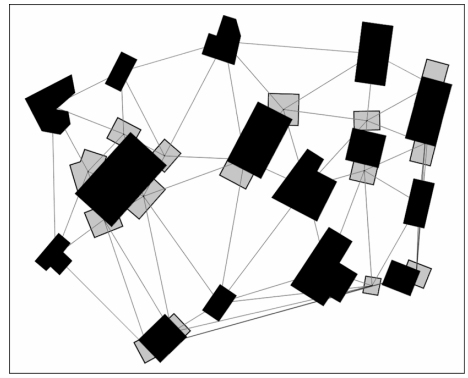


Fig. 11: Typification by collapsing shortest edges (reproduced by permission of swisstopo, BA071153)

The algorithm for building displacement which we developed uses the proximity graph from the triangulation service. This displacement algorithm aims to achieve sufficient distances between the map features so that they are distinguishable and do not visually coalesce. The algorithm can respect different required minimum distances between buildings and between buildings and roads. In an iterative process all edges which are shorter than the minimum distance are requested from the proximity graph. The edge is elongated to the minimum distance which pushes the buildings away from each other. If the proximity edge is between a building and a road segment only the building is moved. After making these modifications the iteration starts again with requesting the edges which are too short. If multiple buildings are very close to each other this process can last a couple of iterations as the buildings once pushed away from each other get again displaced by other buildings. The process stops when no more edges are found that are too short. If not enough space is available to sufficiently separate all features a deadlock may occur. Therefore a maximum number of iterations can be defined. If this limit is reached, the number of remaining short edges and their length can be used to evaluate whether the displacement was sufficient or whether the

number of buildings must be reduced e.g. by a further typification operation. The result of this displacement operator is shown in Figure 13.

Fig. 12: Proximity graph before displacement, (road symbols not to scale; reproduced by permission of swisstopo, BA071153)

Fig. 13: Proximity graph after displacement (road symbols not to scale; reproduced by permission of swisstopo, BA071153)

The WebGen framework allows the use of multiple services with each other. In our implementation the two operator services typification and displacement can use the same proximity graph support service for their purposes. This sequence will now be described shortly.

Before the two operator services can use the stateful support service it must be initialised (see also Fig. 9). Therefore, both building and road features are sent to the support service. Internally the service creates the triangulation and proximity graph data structure and saves it persistently using a unique identifier. The two operator services can access now the graph using this identifier. Typically, the number of the buildings is first reduced by typification, followed by a displacement of the buildings to meet the legibility constraints of the minimum separation distance.

As described above the typification service requests the shortest edge between two buildings from the triangulation support service using the identifier to access the correct graph. As the triangulation and proximity graph contains also the roads there could also be shorter edges between buildings and roads but these will be ignored. Thus, merge operations between two buildings across a road are prohibited. The two buildings connected by the shortest edge are merged to a new placeholder. Then requests to delete the two old nodes from the triangulation and to insert the placeholder are sent to the support service. This process continues iteratively until the desired number of buildings have been replaced by placeholders.

After the typification the displacement service uses the same triangulation and proximity graph service. Therefore it uses the same identifier to access the service. As described above the displacement service retrieves a list of all proximity edges that are to short from the support service. The proximity edges are between buildings as well as between buildings and roads (Fig. 12). All the short edges are elongated by changing the positions of the buildings. Roads remain in their positions. The changes are committed back to the proximity graph service which keeps the graph structure up to date.

Finally the buildings represented by the triangulation and proximity graph can be sent back to the calling client who triggered the typification and displacement workflow. The initial triangulation with its proximity graph is shown in Figure 12. The final result is shown in Figure 14.

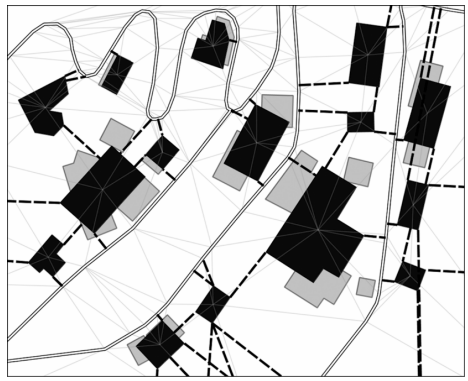


Fig. 14: After typification and displacement (road symbols not to scale; reproduced by permission of swisstopo, BA071153)

This approach of a stateful triangulation service shows how multiple transfers of massive amounts of data between services can be avoided. The building and road features are only transmitted once to the triangulation support service and can be used by multiple operator services. Through the continuous query and update calls this data store remains synchronized. In a scenario with several processing and evaluation steps this same support service can be used throughout the entire workflow. The proximity relations need only be established once and are then continuously modified.

5.2 Detection of Building Alignments

Another enriching structural knowledge are alignment relations as they exist, for example, between buildings. Alignments basically express a special neighbourhood relation. As an algorithm for the detection of alignments an approach inspired by Burghardt and Steiniger (2005) is implemented in the WebGen framework. This algorithm uses principal component analysis to derive the homogeneity of building alignments along roads. Therefore the candidate buildings along every road are characterised using measures about their size, shape, orientation and about their group characteristics (distances to neighbours etc.). Figure 15a shows the homogeneity of building alignments. The darker the colour of the building the more homogeneous is the alignment the building belongs to. Certain buildings such as corner buildings may also be part of several alignments.

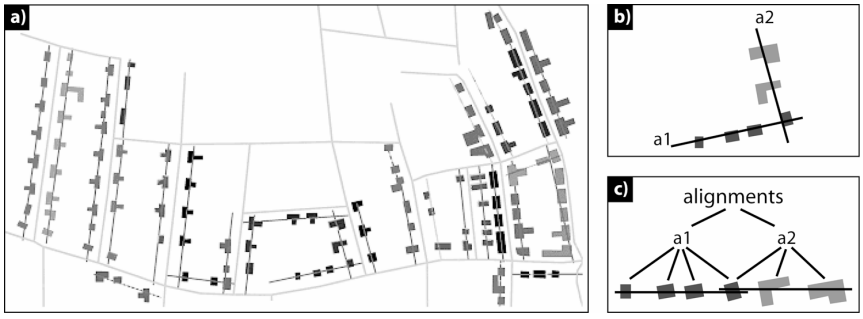


Fig. 15: a) Building alignments (darker buildings are more similar and more strongly aligned). b) Corner buildings are shared by two alignments. c) The root node links all alignments in one data set together. (Reproduced by permission of swisstopo, BA071153)

The knowledge about these alignments may be represented in several ways. Probably the simplest way would be that the alignment support service delivers each alignment group as a

separate set of buildings. Another possibility would be to assign an attribute to the original buildings containing e.g. an identifier for each alignment group and its homogeneity value. These two approaches have the disadvantage that they loose important information such as that a building can be part of more than one alignment. Thus to regain this lost knowledge costly matching procedures might become necessary afterwards.

Using a graph-like representation that is structurally comparable to proximity graphs does not have this disadvantage. Every building can be linked to its neighbours in the same alignment(s). Such a graph can be traversed starting at the root node. This allows a complete representation of all buildings in a data set with their alignment relations. Corner buildings are instantaneously recognisable as they are part of two alignments. An example can be seen in Figure 15b, where the two alignments ‘a1’ and ‘a2’ share a corner building.

The support service for the detection and management of building alignments is implemented as a stateful support service. This service builds up an internal data structure with buildings as nodes, and the alignments as their parent nodes. The root node links all the alignments in one data set (Fig. 15c). This alignment support service can be initialised with buildings and roads. Once initialised, several possibilities exist for query and modification. It is possible to retrieve all alignments as separate groups but it is also possible to query the neighbours of a single building. For example, the typification algorithm (§ 5.1) could be extended to check whether a building is part of an alignment. Also new buildings can be inserted and their alignment relations with the other buildings established. Hence this support service offers both, a simple possibility to merely retrieve alignment groups and more complex capabilities to query individual properties or to modify the alignment structure. This non-hierarchical alignment data structure provides structural knowledge that is flexibly usable throughout a context-dependent generalisation process, restraining for example the typification and displacement of buildings in order not to destroy the existing patterns.

5.3 Attribute and Schema Transformation Services

When generalising maps very often the attribute schemata between the source and the target scale are changing. Attributes may be removed or combined on the target scale, or attribute values may obtain a different domain. A typical example for such an attribute change is the change of feature classes as it occurs when generalising land cover data. According to Hampe et al. (2003) a generalisation process for land cover consists of three steps: reclassification (schema transformation) of the object types; aggregation of adjacent areas with equal object type; and shape generalisation.

The reclassification needs some sort of rule base to inform the assignment of the input features to the target classes. This rule base in many cases is a strict hierarchy that may be represented by a similarity tree which assigns a target class to every input class (e.g. 'deciduous forest' and 'coniferous forest' are both reclassified to 'forest'). In more complex cases the rules must respect the semantical, spatial and topological context. These reclassification rules form a non-hierarchical decision tree which assigns a target class for example only if the feature has a certain attribute, a certain size or is adjacent to a certain feature class (e.g. 'deciduous forest' only becomes ‘forest’ if it is large enough to be visible on the target scale or if it is adjacent to another ‘forest’). Owing to the different requirements, the provision of reclassification rules should also be made by different support services, that is, services for hierarchical static similarity trees (§ 5.3.1) and services that use a context-dependent decision graph (§ 5.3.2).

5.3.1 Hierarchical Reclassification using a Similarity Tree Support Service

A similarity tree or reclassification hierarchy can be defined by the user of the system or generated automatically, for instance by a statistical evaluation of the properties of the input categories to establish their relevance as it was done by Fuchs (2002) using multivariate methods for deriving a statistical aggregation hierarchy of soil types. Thus two types of support services for providing such a similarity tree are possible. The more complex service generates the similarity tree out of the raw data which it receives as input. The simple service just provides a static similarity tree which has been defined by a user. A similarity tree can be modelled using the hierarchical nested structure of XML (§ 4.2). Such an XML file (see Listing 1) can easily be queried in order to perform a reclassification. Typically the reclassification service iterates over a set of input features. For every

feature it derives the parent class of the current feature class (XML query getParent()) and assigns it to the feature. Hence every 'deciduous forest' or 'coniferous forest' feature would become 'forest' after that step. The feature classes are usually stored as attributes with the land cover features. Thus, for every feature the reclassification changes the value of this class to its parent class. For a typical reclassification service this similarity tree could be requested as an XML hierarchy from a support service, defined either by a user or generated automatically. For user-generated similarity trees the support service acts just as file server, delivering the static hierarchy. Automatically generated similarity trees are inferred from the input data using e.g. statistical methods.

For testing purposes two examples have been implemented. First, a static similarity tree support service which serves an XML hierarchy (see Listing 1) to reclass land cover data from 28 different classes to 12 user-defined classes. Second, a dynamic similarity tree service that generates automatically a hierarchy based on the distribution of the feature classes. For distribution based reclassification three methods have been implemented: quantiles; preserve small classes; and preserve large classes. These methods are just simple examples for testing purposes. More advanced methods like the Hierarchical Clustering proposed by Fuchs (2002) or statistical methods like natural breaks or quantiles could be implemented the same way to generate a similarity tree. It is not only possible to retrieve the whole similarity tree but also to query the corresponding parent class for a given class. This service can be used if only few features are processed. So, other services can request similarity tree information without having to care about the decoding of the XML hierarchy and without having to download the whole similarity tree.

Figure 17 shows the result of the reclassification and merging step. In the reclassification step the original 28 land cover classes (Fig. 16) are reduced to 12 classes. The blank areas are not empty; they are occupied by the class 'other'. In the merging step adjacent polygons having the same class were combined.

Fig. 16: Land-cover data 28 classes (reproduced by permission of swisstopo, BA071153)

Fig. 17: Land-cover data 12 classes (reproduced by permission of swisstopo, BA071153)

5.3.2 Non-hierarchical (Conditional) Reclassification and Attribute Schema Transformation

In the reclassification example above a pure hierarchy is used to modify the class attribute of every feature. In this example a more complex approach is demonstrated which can change feature attributes, including the class attribute, based on rules with conditions.

As can be seen in Figure 17 some of the small features are assigned to the 'other' feature class but it would have been better if these features got assigned to the class of a neighbouring feature. Thus, in order to perform reclassifications or other processes where attributes change or features get assigned a new class or merged, a rule-based approach can be helpful. Figure 18 shows a simple decision graph with some condition checks which decide the assignment of a new feature class.

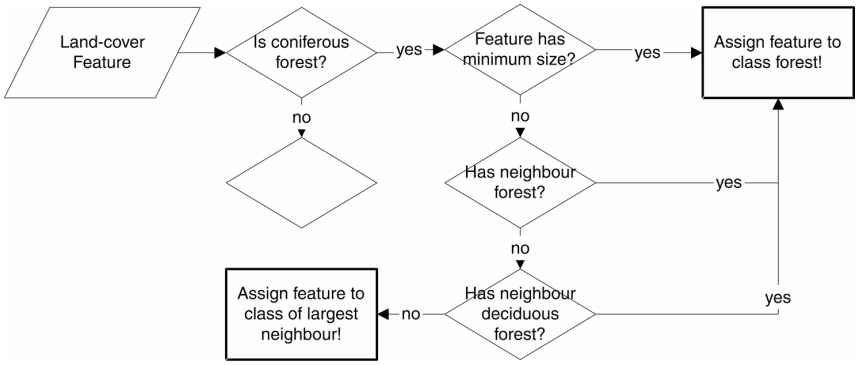


Fig. 18: Conditional decision graph

Transformation rules for feature classes and attributes can be subdivided into semantical, geometrical and structural (topological) rules (Bobzien, 1999). Semantical rules are simply checking attribute values of the source features, whereas geometrical and structural rules are evaluating properties and the context of a feature. In the WebGen framework the geometrical and structural properties are provided as measures by support services.

The implemented example of a rule base service is used for the conditional reclassification. Therefore it is intended to apply transformation rules onto the supplied features. Bobzien (1999) proposes therefore a rule syntax which can be seen in the rules column of Listing 2. These transformation rules are loaded into the support service where they are decoded and applied.

source class	*target class*	*attribute*	*target value*	*rules*
coniferous forest	forest			#size >= 200
coniferous forest	forest			#touches 'forest'
coniferous forest	forest			#touches 'deciduous forest'
coniferous forest	farmland			#size < 200 AND #touches 'farmland'
	farmland	cultivation	coniferous	
coniferous forest	other			#size < 200 AND #touches 'other'
…	…	…	…	…

Listing 2: Conditional reclassification and attribute schema transformation rules

Geometrical and structural properties here are preceded by a pound sign. Currently simple geometrical properties such as #size, #length or #width are implemented in WebGen as well as the structural property #touches. Rules that change the class like the transformation from coniferous forest to farmland can be followed by additional rules and assignments of attributes from the source onto attributes of the target schema. Similar conditional services could also be of use for schema transformations in model generalisation or for a building typification algorithm which respects e.g. semantical rules for the generation of the placeholder.

5.4 Urban Structure Classification

An example for a very costly support service, implemented in the WebGen framework, is the classification of buildings according to their urban structure. In Steiniger et al. (in press) five types of urban structures were defined. The knowledge whether a building is in an industrial and commercial area, in a inner city area, in a urban area (dense buildings), in a suburban area (dispersed buildings) or in a rural area (single buildings) can be used for different purposes. Examples are different colourings or the amalgamation of buildings in dense areas and the preservation of single buildings in less dense areas (see examples in Fig. 19).

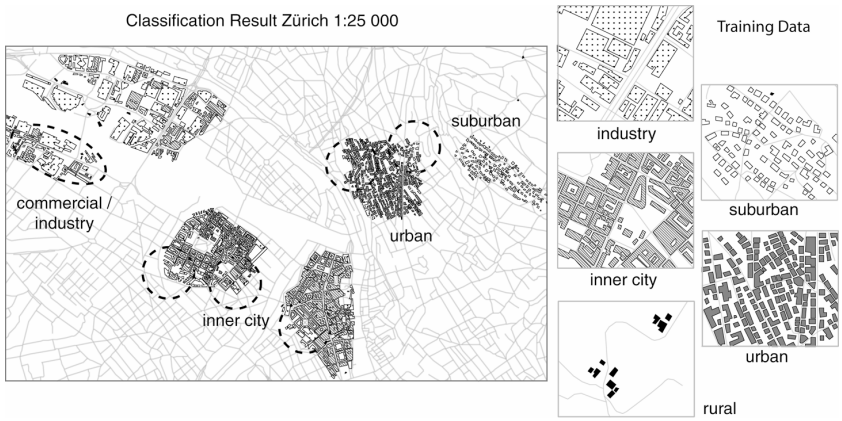


Fig. 19: Urban structure types (Steiniger et al., in press; reproduced by permission of swisstopo, BA071153)

5.4.1 Classification as Enriching Attributes

The approach of Steiniger et al. (in press) for the classification of urban structures uses solely geometrical and perceptual measures of buildings (see process sequence in Fig. 20). Morphological measures that are used include the building area; the number of building corners; the building shape; the building squareness; the building elongation; and the number of courtyards. Relational measures encompass the number of buildings intersected by a buffer; the building area in relation to the convex hull area of the buffer intersecting buildings; and the relation of the building area to the buffer area of all intersecting buildings. When using all these measures for characterising the buildings they are spanning a 9-dimensional space (one dimension per measure) and the borders separating the urban structure types must be defined in this full 9-D feature space.

The implemented classification support service uses the Batch-Perceptron approach (Duda et al., 2000) to derive the separating decision borders (thresholds) using training data where the urban structure types are known. These thresholds are represented as a matrix which describes the relations between different shape measures of the building features. Following this training step in the classification step the buildings that should be classified are first characterised applying the measures. Then the urban structure types are derived using the decision borders and assigned to the building features as attributes.

The advantage of this service is its relative simplicity. Training data and the data to classify have to be sent to the service. As result the data to classify are sent back with two attributes assigned to every feature denoting its urban structure type and the accuracy of this classification. This enriching information can easily be used by various generalisation operators. The disadvantage is that especially the training process in order to derive the decision borders is time-consuming.

5.4.2 Separate Characterisation, Training and Classification

In order to increase the flexibility and also the performance of the extraction of the urban structure types the whole process was subdivided into components. The characterisation of the training data using measures and the derivation of the decision borders in the training process has to be carried out only once and the result can be reused. The characterisation of the input data using the measures has to be done for every dataset which has to be classified. During the characterisation process a number of measures are calculated for every feature positioning it with its characteristics in a multi-dimensional space. The calculation of every measure can be accomplished by a separate support service (Fig. 20) which takes the dataset as input and delivers a list with the measures for every feature in the dataset.

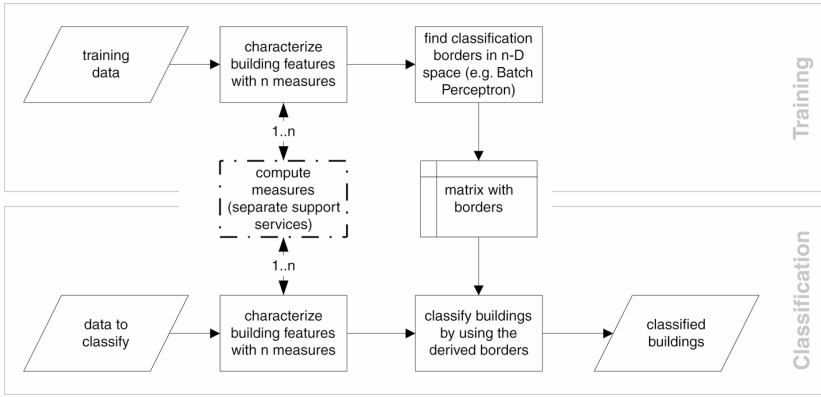


Fig. 20: Classification steps (training, characterisation and classification)

The training and classification processes can both work with an arbitrary number of dimensions of the feature space. Thus, due to the flexible service architecture it is possible to dynamically select the measures that are used in the characterisation, allowing to tune the result of the classification. Some measures like the perceptual measures are quite heavy to calculate. Support services for the measures can reside on different servers and the load can thus be distributed for parallel execution.

The output of the training process is a matrix. The number of columns of this matrix is simply the number of measures that are used to characterize the features. The number of columns is the number of all possible combinations of two different classes (10 in the case of 5 urban structure classes). This matrix is represented and stored like a table using XML. The training support service, implemented as a stateful service (see § 4.4), can store the matrix with the decision borders persistently. The classification service itself requests this matrix and performs the classification. Thus the training process has only to be done once for a specific data type and can then be used various times, saving a lot of time. Together with the possibility to carry out the characterisation of the buildings in parallel on separate servers this offers a great performance improvement potential.

6. Discussion

Many algorithms for enriching data or creating spatial relationships are only available on different standalone platforms and data formats. Thus interoperability and reusability are not ensured, the different implementations of algorithms are not comparable and implementations get even lost with time, particularly in academic research environments (Regnauld, 2005). The development process of advanced generalisation operators takes more time because algorithms or converters for auxiliary data structures have to be generated first.

It is important to note that today most supporting data structures are only generated at runtime and not saved persistently. Some generalisation support structures are very expensive to calculate but could easily be saved persistently for multiple uses, for instance, by other generalisation operators. The approach of stateful support services such as the triangulation service presented in § 5.1 reveals that due to the large amount of calls to a service performance problems might occur due to network latencies. Thus it must be evaluated for every generalisation process at which degree of granularity the subdivision into smaller or larger calls and components is appropriate.

Another support structure which is usually not saved persistently and only visible inside a process are trained models produced by machine learning approaches. Support services could also be used to keep trained models persistent and make them available to other services. However, this must be done with regard to the learning approach used and to the portability of the information learned.

The aim of this paper was to illustrate the possible usage of generalisation support services as web services. The WebGen framework implementation shows a great potential for enabling the

interoperable, flexible and reusable deployment of generalisation algorithms. Many of the concepts do not only apply to generalisation but also too many other tasks where spatial data are used in a distributed, heterogeneous network environment. The WebGen architecture is not intended to be a web service for managing, browsing and exchanging data over the Internet but a *processing service* for providing algorithm logics and processing power to be executed remotely. It is, in contrast to other commercial or non-commercial GIS geo-processing servers, not limited to a (proprietary) GIS platform. The usage of an open protocol and standards such as XML and HTTP enable the interoperable coupling of service providers and consumers.

The use of web services for providing supporting data structures might not always be appropriate: when working with extremely large data volumes or high frequencies of service invocations the process performance might be poor due to network latency, bandwidth and the time needed for creating and parsing the XML messages. For a single service use always the whole data has to be transferred from the calling client to the service and back. For multiple service invocations a stateful approach like the transaction server (see §4.4) can reduce the effectively transferred amount of data.

Developers wanting to use a supporting data structure do not have to re-implement or integrate source code of others; they still need to write a parsing routine to read the XML graph structure, for example. This is clearly a hurdle that must be taken, but as the web services are respecting common interfaces, different support services providing different graph structures, for instance, can then be used without having to recreate the entire parser.

Standardized functionalities to compute and use spatial and structural relationships are sparse in GIS packages, particularly in commercial ones. Furthermore, geographic databases or data transfer formats do not commonly include the modelling and storage of the advanced relationships discussed above. We proposed ways to use this knowledge in a web services environment using XML. What is still missing today, however, is an agreement in the generalisation community on how to exploit these XML-based formats to achieve a standardised data format for representing the generalisation support structures, many or most of which are graph data structures. Such a format can be used in the development of new generalisation support services and possibly also converters for already existing support structures could be developed. Following the categories of generalisation services (see § 2.4) the generalisation support services can then form the foundation of the more advanced generalisation operator and generalisation process services.

7. Conclusion

We distinguish three types of web services necessary for a comprehensive service-based generalisation architecture, generalisation support services, generalisation operator services, and generalisation process services. In this paper, we dwelled primarily on the generalisation support services, which are intended to enrich the raw cartographic input data with additional information such as shape or importance attributes, new geometries, as well as spatial and structural relationships, hence providing indispensable support to the other two service categories. These support services could be equally useful in a more general, non-generalisation environment, particularly for the purposes of spatial analysis and decision support.

As a first contribution, this paper delivered a comprehensive taxonomy of generalisation support services in relation to different generalisation operators. Second, methods were proposed to represent, store and exchange the spatial relations generated by support services, considering the special requirements of distributed architectures. Many relations can be expressed in a graph-like form. Thus, the proposed data structures and formats are mainly graph based. Third, we have presented implementation examples of generalisation support services in the WebGen generalisation service framework. It was demonstrated how generalisation support services can interact with generalisation operator services, delivering complex data structures to complex algorithms. Obviously, as mentioned in the Discussion, there are still plenty of open problems remaining, as the delivery of generalisation capabilities is revisited, facilitated by a different architecture (service-based rather than standalone) and at least partially given more stringent constraints w.r.t. processing efficiency.

We plan to address the above issues in the future by continuously extending the WebGen platform. Given the extensible nature of the service-based approach the WebGen framework lends itself to the collaboration with other researchers, which has occasionally already happened. Currently, attempts are made to combine support services and operator services with process services, in order to achieve automated control of the generalisation workflow.

Acknowledgements

This research was partially funded by the Swiss National Science Foundation (grant 20-101798, project DEGEN). Thanks go to Alistair Edwardes, Ingo Petzold and Matthias Bobzien for helpful comments.

References

1SPATIAL, 2007, Radius Clarity, http://www.1spatial.com (accessed 03/2007).

ANDERS, K.-H., 2004, Graph-Clustering-Verfahren zur Interpretation raumbezogener Daten. PhD thesis, Institute of Photogrammetry, University of Stuttgart.

ARMSTRONG, M. P., 1991, Knowledge classification and organization. In Buttenfield, B. P., McMaster, R. B. (eds), *Map Generalization: Making Rules for Knowledge Representation*, Longman, 86-102.

BADER, M. and WEIBEL, R., 1997, Detecting and Resolving Size and Proximity Conflicts in the Generalization of Polygonal Maps. *Proceedings 18th International Cartographic Conference,* Stockholm (S), 1525-1532.

BADER, M., BARRAULT, M. and WEIBEL, R., 2005, Building displacement over a ductile truss. *International Journal of Geographical Information Science*, 19, 915–936.

BARRAULT, M., N. REGNAULD, C. DUCHÊNE, K. HAIRE, C. BAEIJS, Y. DEMAZEAU, P. HARDY, W. MACKANESS, A. RUAS and R. WEIBEL. 2001. Integrating multi-agent, object-oriented and algorithmic techniques for improved automated map generalization. *Proceedings of XX Int. Cartographic Conference*, Beijing, pp. 2110–2116.

BOBZIEN, M., 1999, Implementationsaspekte der Modellgeneralisierung, In *Mitteilungen des Bundesamtes für Kartographie und Geodäsie*, Vol. 3.

BOBZIEN, M., BURGHARDT, D., PETZOLD, I., NEUN, M. and WEIBEL, R., 2006, Multi-Representation Databases With Explicitly Modelled Intra-Resolution, Inter-Resolution and Update Relations. In *Proceedings of the American Congress on Surveying and Mapping, AutoCarto*.

BOBZIEN, M. and MORGENSTERN, D., 2002, Geometry-Type Chance in Model Generalization - a Geometrical or a Topological Problem?. In *Proceedings of the Joint ISPRS/ICA Workshop on Multi-Scale Representations of Spatial Data*, Ottawa.

BURGHARDT, D. and CECCONI, A., 2007, Mesh Simplification for Building Typification. *International Journal of Geographical Information Science*, 21(3): 283-298.

BURGHARDT, D. and NEUN, M., 2006, Automated sequencing of generalisation services based on collaborative filtering. In *Geographic Information Science*, M. Raubal, H. Miller, A. Frank, M. Goodchild (Eds) (Springer, LNCS 4197) pp. 41-46.

BURGHARDT, D., M. NEUN and R. WEIBEL, 2005, Generalization Services on the Web – A Classification and an Initial Prototype Implementation, *Cartography and Geographic Information Science*, 32(4), 257-268.

BURGHARDT, D. and STEINIGER, S., 2005, Usage of Principal Component Analysis in the Process of Automated Generalisation. In *Proceedings of the XXII International Cartographic Conference*, A Coruña, Spain, CD-ROM.

BUTTENFIELD, B. P., MCMASTER, R. B., (1991, eds), *Map Generalization: Making Rules for Knowledge Representation*. London, Longman.

CHANNABASAVAIAH, K., HOLLEY, K. and TUGGLE, E., Migrating to a service-oriented architecture. *IBM DeveloperWorks*, http://www-128.ibm.com/developerworks/library/ws-migratesoa/ (Accessed 07/2006).

CHITHAMBARAM, R., BEARD, K. and BARRERA, R., 1991, Skeletonizing Polygons for Map Generalization. *Technical Papers ACSM*, Baltimore, Vol. 2, pp. 44 - 55.

CHRISTOPHE, S. and RUAS, A., 2002, Detecting Building structures for generalisation purposes. In *Advances in Spatial Data Handling*, D. Richardson and P. van Oosterom (Eds) (Berlin, Springer), pp. 419–432.

DE FLORIANI, L. and PUPPO, E., 1995, Hierarchical triangulation for multiresolution surface description. *ACM Transactions on Graphics*, 14, 363–411.

DUDA, R., HART, P., and STORK, D., 2000, *Pattern Classification*. 2nd edition, John Wiley, New York.

EDWARDES, A., BURGHARDT, D., BOBZIEN, M., HARRIE, L., LEHTO, L., REICHENBACHER, T., SESTER, M. and WEIBEL, R., 2003, Map Generalisation Technology: Addressing the need for a common research platform. In *Proceedings of the Proceedings of the 21st International Cartographic Conference*, Durban, South Africa.

EDWARDES, A., BURGHARDT, D. and NEUN, M., 2007, Experiments to build an open generalisation system. In *Generalisation of geographic Information: Cartographic Modelling and Applications*, W. Mackaness, A. Ruas and T. Sarjakoski (Eds) (Elsevier Science).

ESRI, 1998, ESRI Shapefile Technical Description, http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf.

FITZKE, J., GREVE, K., MÜLLER, M. and POTH, A., 2004, Building SDIs with Free Software the deegree project. In *Proceedings of the 7th Conf. Global Spatial Data Infrastructure*, Bangalore, India.

FUCHS, M., 2002, Methoden zur objektiven Ableitung von Bodenkarten im Folgemastab. PhD thesis, Free University of Berlin.

HAMPE, M., ANDERS, K.H. and SESTER, M., 2003, MRDB Applications for Data Revision and Real-Time Generalisation. In *Proceedings of the Proceedings of 21st International Cartographic Conference*, August, Durban/South Africa.

HARRIE, L. and WEIBEL, R., 2007, Modelling the Overall Process of Generalisation. In *Generalisation of Geographic Informations: Methods and Applications*, W. Mackaness, A. Ruas and T. Sarjakoski (Eds) (Elsevier Science).

HAUNERT, J.-H. and SESTER, M., 2004, Using the Straight Skeleton for Generalisation in a Multiple Representation Environment. *ICA Workshop on Generalisation and Multiple Representation,* Leicester (UK), http://ica.ign.fr/Leicester/paper/Haunert-v2-ICAWorkshop.pdf, (accessed 01/2007).

HOLT, R., WINTER, A. and SCHÜRR, A., 2000, GXL: Toward a Standard Exchange Format, http://www.gupro.de/GXL/ (accessed 07/2006).

HORTON, H., 1945, Erosional Development of Streams and their Drainage Basins. *Bulletin of the Geological Society of America*, 56, 275–370.

ICA, 2004, Brain storming Sessions at the ICA Workshop on Generalisation and Multiple Representation, available from http://ica.ign.fr/.

ILLERT, A. and AFFLERBACH, S., 2004, Global schema specification, GiMoDig-project, Deliverable D5.3.1, Public EC report, http://gimodig.fgi.fi/deliverables.php (accessed 12/2006).

JONES, C.B., BUNDY, G.L. and WARE, J.M., 1995, Map Generalization with a Triangulated Data Structure. *Cartography and Geographic Information Systems*, 22, 317–331.

JUMP, 2007, The JUMP Unified Mapping Platform, http://www.jump-project.org (accessed 03/2007)

LEHTO, L. and SARJAKOSKI, T., 2004, An Open Service Architecture For Mobile Cartographic Applications. In *Proceedings of the Location Based Services & TeleCartography*, G. Gartner (Ed.).

LÜSCHER, P., 2005, Matching von Strassendaten stark unterschiedlicher Maßstäbe und Aufbau einer MRDB. In *Sitzung Arbeitsgruppe Automation in Kartographie, Photogrammetrie und GIS*, Wien.

MACKANESS, W.A. and BEARD, M.K., 1993, Use of Graph Theory to Support Map Generalization. *Cartography and Geographic Information System*, Vol. 20, No. 4, pp. 210–221.

MCMASTER, R. and SHEA, S., 1992, *Generalization in Digital Cartography* (Washington, USA: Association of American Geographers).

MUSTIÈRE, S. and MOULIN, B., 2002, What is Spatial Context in Cartographic Generalisation?, *Conference on Geospatial Theory, Processing and Applications, IAPRS & SIS*, 34(4) 274-278.

NEUN, M. and BURGHARDT, D., 2005, Web Services for an Open Generalisation Research Platform. In *Proc. 8th ICA Workshop on Generalisation and Multiple Representation*, A Coruña, http://ica.ign.fr/.

NEUN, M., WEIBEL, R. and BURGHARDT, D., 2004, Data Enrichment for Adaptive Generalisation. In *Proc. 7th ICA Workshop on Generalisation and Multiple Representation*, Leicester, http://ica.ign.fr/.

OGC. 1999, OpenGIS® Simple Features Implementation Specification for SQL, http://www.opengeospatial.org/specs/ (accessed 07/2006).

OGC, 2002, The OpenGIS® Abstract Specification, Topic 12: OpenGIS Service Architecture, Version 4.3, OGC 02-112. http://www.opengeospatial.org/specs/ (accessed 07/2006).

OGC, 2004, The OpenGIS® Geography Markup Language (GML) Encoding Specification, Version 3.1.1, OGC 03-105r1. http://www.opengeospatial.org/specs/ (accessed 07/2006).

OGC, 2005, The OpenGIS® Discussion Paper: Web Processing Service (WPS), Version 0.4, OGC 05-007r4. http://www.opengeospatial.org/specs/ (accessed 07/2006).

PETZOLD, I., BURGHARDT, D., and BOBZIEN, M., 2005, Automated derivation of town plans from large scale data on an example of area to line simplification. In *Proceedings of the XXII International Cartographic Conference*, A Coruña, Spain, CD-ROM.

PFALTZ, J.L., 1976, Surface Networks. *Geographical Analysis*, 8, 77–93.

PLAZANET, C., AFFHOLDER, J.G. and FRITSCH, E., 1995, The Importance of Geometric Modeling in Linear Feature Generalization. *Cartography and Geographic Information Systems*, 22, 291–305.

RANA, S., 2004, *Topological Data Structures for Surfaces: An Introduction to Geographical Information Science*. John Wiley.

REGNAULD, N., 2005, Spatial Structures to Support Automatic Generalisation. In Proceedings of the *Proceedings XXII International Cartographic Conference*, A Coruña, Spain, CD-ROM.

REGNAULD, N., 2006, Improving Efficiency for Developing Automatic Generalisation Solutions. In Proc. *Joint ISPRS Workshop on Multiple Representation and Interoperability of Spatial Data*, Hannover.

RUAS, A., and C. PLAZANET. 1996. Strategies for automated generalization. In: *Proceedings 7th International Symposium on Spatial Data Handling (Advances in GIS Research II)*, Taylor & Francis, London, pp. 6.1–6.17.

RUAS, A., 1998, A method for buiding displacement in automated map generalisation. *International Journal of Geographical Information Science*, 12, 789–803.

RUAS, A., 2000, The Roles of Meso Objects for Generalisation. In *Proceedings 9th Symposium on Spatial Data Handling*, Beijing, China, pp. 3b50–3b63.

SARJAKOSKI, T., SESTER, M., SARJAKOSKI, L., HARRIE, L., HAMPE, M., LEHTO, L. and KOIVULA, T., 2005, Web generalisation services in GiMoDig - towards a standardised service for real-time generalisation. In *Proceedings of the 8th AGILE Conference on GIScience*, F. Toppen and M. Painho (Eds), Estoril, Portugal.

SHEWCHUK, J., 1996, Triangle: Engineering a 2-D quality mesh generator and Delaunay triangulators. In *Proceedings first workshop on Applied Computational Geometry*, PA, USA.

STRAHLER, A.N., 1952, Hypsometric (Area-Altitude) Analysis of Erosional Topology. *Bulletin of the Geological Society of America*, 63, 1117–1142.

THOMSON, R. and RICHARDSON, D., 1995, A graph theory approach to road network generalization. In *Proceedings of the 17th ICA Meeting*, Barcelona, Spain.

TIMPF, S., 1998, Hierarchical Structures in Map Series. PhD thesis, Technical University of Vienna.

VAN OOSTEROM, P., 1993, *Reactive Data Structures for Geographic Information Systems*, Oxford University Press.

VAN OOSTEROM, P. and SCHENKELAARS, V., 1995, The development of a multi-scale GIS. *International Journal of Geographical Information Systems* **9**(5): 489-508.

VAN SMAALEN, J., 2003, Automated Aggregation of Geographic Objects: A New Approach to the Conceptual Generalisation of Geographic Databases. PhD thesis, Wageningen University and Research Centre.

WARE, J., JONES, C. and THOMAS, N., 2003, Automated Map Generalization with Multiple Operators: A Simulated Annealing Approach. *International Journal of Geographical Information Science*, 17, 743–769.

WORBOYS, M.F. and DUCKHAM, M., 2004, *GIS. A Computing Perspective*, 2nd edition, CRC Press.

Research Paper 4

Neun M., D. Burghardt and R. Weibel (submitted): Automated Processing for Map Generalization with Modular Operator Services. Submitted to *GeoInformatica*.

Automated Processing for Map Generalization with Modular Operator Services

Moritz Neun*, Dirk Burghardt, Robert Weibel

University of Zurich, Department of Geography, Winterthurerstrasse 190, CH-8057 Zurich, Switzerland

Abstract: In map generalization various operators are applied to the features of a map in order to maintain and improve the legibility of the map after the scale has been changed. These operators must be applied in the proper sequence and the quality of the results must be continuously evaluated. Cartographic constraints can be used to define the conditions that have to be met in order to make a map legible and compliant to the user needs. The combinatorial optimization approaches shown in this paper use cartographic constraints to control and restrict the selection and application of a variety of different independent generalization operators into an optimal sequence. Different optimization techniques including hill climbing, simulated annealing and genetic deep search are presented and evaluated experimentally by the example of the generalization of building blocks. All algorithms used in this paper have been implemented in a web services framework. This allows the use of distributed and parallel processing in order to speed up the search for optimized generalization operator sequences.

Keywords: map generalization, data enrichment, cartographic constraints, combinatorial optimization, parallel processing

1 Introduction

Map generalization seeks to maintain and improve the legibility of a map after the scale has been changed. During the map generalization process various generalization operators are applied to the features of a map such as buildings, roads, rivers or land cover patches. Examples of such generalization operators include algorithms for the simplification, smoothing, aggregation, amalgamation, merging, collapse, refinement, exaggeration, enhancement and displacement of cartographic features [19]. These operators must be applied in the optimal sequence, with the correct parameterization and their results must be evaluated in order to achieve an improvement of the map legibility in the generalization process. Thus, if the aim is to fully automate the map generalization process and minimize human intervention, there is a need for automated control and sequencing of generalization operators.

Several conditions have to be met in order to make a map optimally legible. These conditions can be formalized with so called cartographic constraints [38]. Figure 1 shows an example of conflicts that arise during map generalization. The left picture shows the conflicts that are created when the width of road symbols is enlarged. The buildings then overlap with the road symbols. Furthermore, some of the buildings are too small and would no longer be visible at the target scale (right picture). Therefore different generalization operators must be applied to resolve these conflicts, and they must be applied in the proper sequence. The operators that lead to the result in Figure 1 are building simplification, typification and displacement, in this order.

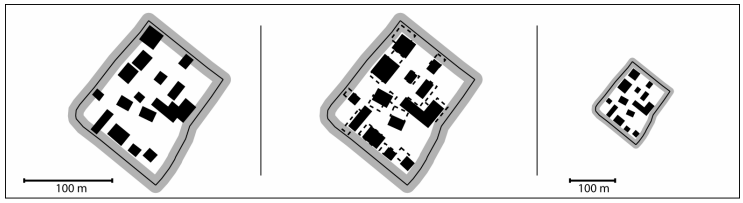


Fig. 1: Conflict due to road symbolization, solved through simplification, deletion and displacement (examples show the source scale, the generalized result at the source scale and at the final target scale)

The methods presented in this paper use cartographic constraints for the selection and sequencing of different independent generalization operators. Manifold independent algorithms with different grades of sophistication (from simple algorithms to sophisticated, context-aware algorithms) can be combined

* Corresponding author. E-mail address: moritz.neun@geo.uzh.ch

to jointly form a powerful workflow by uniting their strengths. We use a modular service based architecture which allows the integration of different stand-alone generalization operators even from different generalization systems on different platforms ([7],[22]). With this versatile service based architecture it is possible to build modular generalization processes that can flexibly be extended by introducing additional generalization operators. Thus, having a relatively large set of different operators available as services the goal is to select the optimal operator sequence in order to reduce the constraint violations for a given problem. This calls for the use of combinatorial optimization techniques [16] which control and restrict the application of the various operators.

The focus of this work is on introducing and revising constraint based combinatorial optimization techniques for the control of the generalization process. The goal is not to optimize a specific generalization task using one specific algorithm but to find and optimize a sequence of multiple generalization operators which are applied to an entire set of map features. In our case we apply stand-alone generalization operators onto complete building partitions derived from a trans-hydro-graph [33]. Thereby it must be taken into account that the applied operators can have a different granularity in terms of their behavior. Some operators induce only small changes, or changes that can always be undone, while others make quite radical and irreversible changes (see §3.4). We show that it is possible to create an automated generalization workflow that applies stand-alone generalization operators in an optimized sequence. The service based architecture allows the coupling of the operator services together with supporting facilities, like the evaluation functions, for being processed with arbitrary optimization strategies. This novel modular processing approach also makes it possible to use parallelization techniques in order to speed up computationally heavy processing.

After introducing cartographic constraints and reviewing current automated generalization approaches (§2) the paper describes the constraints and the cost function used in the optimization techniques (§3). In § 4, the different optimization techniques including exhaustive search, hill climbing, and genetic search are explained. In §5 the implementation of the workflow control in the services based architecture is described and the improvement of the processing performance using parallelization is demonstrated. Finally results of the tests of the different optimization techniques are presented (§6) and discussed (§7). The paper ends with a conclusion (§8).

2 Background

2.1 Cartographic Constraints

Conventional and automated map generalization share the same basic objective, which is to ensure the legibility of a map for the map reader. Conditions that have to be met in order to make the map legible can be formalized with so called constraints. Examples of cartographic constraints are those listed in §3. The concept of cartographic constraints was originally adapted from computer science to map generalization by [5]. Constraints received special importance in cartography through the application of intelligent agents in the area of automated generalization [28]. Following the results from AGENT project ([4],[29]) constraints define a final product specification on a certain property of an object that should be respected by an appropriate generalization. While measures only characterize objects or situations [27], without considering cartographic objectives, constraints evaluate situations with respect to the formalized cartographic objectives. Thus the constraints check whether the objects or situations are also in a cartographically satisfactory state.

Constraints can be used to describe object characteristics and relationships according to the requirements for a specific map scale and type. [3] proposed three types of assessment functions to determine the quality of cartographic generalization. The first type are characterization functions, which characterize the geometrical and structural properties of single features or groups of features by means of constraints. The second type are the evaluation functions, which compare the states of the features before and after generalization. The third type includes the aggregation functions, which are used to summarize the individual evaluation results. Similar to this methodology, our approach aims to calculate a global cost function as proposed for the constraint space by [8].

2.2 Automating the Generalization Process

Automated control of the generalization process can be achieved using different approaches. [14] review existing methods, including simple batch processing, condition-action modeling, and finally sophisticated constraint based techniques.

Static generalization workflows can be executed as batch processes. Here, no conditions can be applied and the parameters as well as the sequence of the operators are predefined. The modeling of conditional workflows within a generalization system is shown for example by [23]. For such rule-based processing approaches a human expert must formalize the cartographic knowledge into conditions and thus explicitly defines the relation between conditions and actions and their order of processing. The selection of such rules is addressed for example by [26]. This selection of rules is also part of the knowledge acquisition process [11]. [37] propose machine learning techniques for deriving the cartographic rules. The combination of constraint based evaluation and machine learning techniques for the knowledge acquisition is shown by [21].

Using constraint based techniques the cartographic knowledge is captured in terms of conditions that have to be met. To satisfy these constraints optimization techniques may be used. The goal is always to minimize the violation of the constraints. The constrained based methods can be further subdivided into complex techniques which perform different operations simultaneously, as opposed to methods that use constraints to chain specific algorithms that perform one operation at a time [20]. Examples of the first category are for instance least squares adjustment ([13],[30]), energy minimization ([6],[2]) or simulated annealing [36], whereas the AGENT approach ([28],[24],[29]) belongs to the second one. The AGENT approach tries to minimize the constraint violations for map features represented by autonomous software agents. These agents are trying to find an optimal state with minimal constraint violations. Combinatorial optimization methods [16] also try to find an optimal state. [35] are using iterative improvement for the displacement of buildings. In [36] an iterative improvement approach with building displacement, scaling and deletion is shown. This approach uses simulated annealing in order to find a global constraint violation minimum.

3 Constraints for Evaluating Map Partitions

In this paper the development and implementation of a processing service for the generalization of individual buildings is presented. Therefore the legibility conditions focus exclusively on geometrical and structural aspects. For the characterization and evaluation of the generalization state of a building the following eight constraints were defined (see also Fig. 2):

- the building must have a minimum size in order to be visible (acronym: MinSize; number: c0)
- the edges of the buildings must have a certain length in order to be visible (EdgeLength; c1)
- the buildings must be separated by a certain distance in order to be distinguishable (MinDist; c2)
- the buildings must have no parts which are not visible (LocalWidth; c3)
- the position of the building should be preserved as far as possible (DiffPos; c4)
- the number of edges of the building should be preserved as far as possible (DiffEdgeCount; c5)
- the width/length ratio of the building should be preserved as far as possible (DiffWidthLen; c6)
- the orientation of the building should be preserved as far as possible (DiffOrientation; c7)

Additional constraints could be inserted if needed in order to better define the desired map properties. The selected constraints can be established for every building. The minimum distance constraint (MinDist), however, analyzes the neighborhood of the building, while all other constraints solely focus on the geometry of an individual building. The four “Diff” constraints are responsible for the preservation of feature characteristics. At the initial situation they are satisfied but during the map generalization process the may be violated. For example, the position constraint (DiffPos) is violated when a building is displaced (see §3.4).

3.1 Independent Map Partitions

Partitions subdivide the map space in such a way that generalization tasks can process and evaluate them independently. This is an important preliminary step for the generalization of seamless data sets instead of map sheets. The partitions try to subdivide the data into coherent parts, hence isolating the generalization tasks. The features used as borders of the partitions should be static and should not change much during generalization.

For deriving partitions for building data the trans-hydro-graph can be used as proposed by [33]. This graph describes a structure derived from the transportation and hydrology networks. The trans-hydro-

graph can be used to isolate the task of building generalization since buildings must stay inside the faces of the trans-hydro-graph. The partitions are formed by building blocks and are usually small, containing between 1 and 80 buildings in our examples. Thus, they allow faster and parallelized execution of context dependent generalization algorithms as the data structures used do not become too large. The result of this explicitly established partonomic relation is a list of groups, whereby each group may contain any number of buildings. Thus, a building partition can also be seen as a meso-object [28]. Every partition can be characterized by a number of constraints which define an ideal cartographic situation. The constraints describe the fulfillment of a condition for every feature in the partition. Additional group constraints are describing conditions for the entire partition. The evaluation of a generalization result is carried out for an entire partition; therefore a cost function with weights for the different constraints is used (see §3.3).

3.2 Constraint Visualization with Parallel Coordinate Plots

For representing and analyzing the state of a cartographic situation the violations of constraints can be represented by so called generalization state diagrams ([28],[4]). For the visualization of large numbers of features with their associated constraints we propose to use parallel coordinate plots (Figure 2). The parallel coordinate plots represent *n* cartographic constraints with their degree of satisfaction. The axes are scaled to the interval [0, 1], whereby a value greater zero means that the constraint is violated. The constraint values are equivalent to the so called severity used in the generalization state diagrams.

Figure 2 shows the constraint violations for a building block before generalization. The thin lines in the plot show the constraint violations for every individual building, the heavy line shows the average constraint violation for the entire building block. Note that constraints are not weighted.

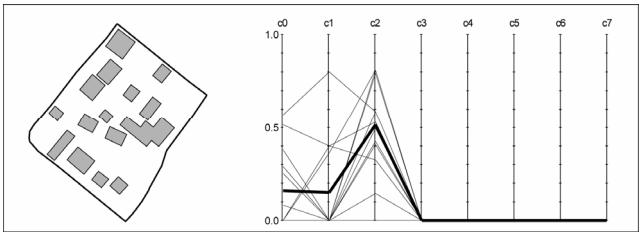


Fig. 2: Constraint violation before generalization

Figure 3 depicts the constraint violations after generalization. This example shows that the generalization process has generally reduced the violation of the various constraints. In comparison with Figure 2 it is also noticeable that in order to reduce the violation of constraints c0 to c3 the preserving constraints c4 to c7 had to be violated. The importance of the individual constraints and thus a possible weighting is expressed by means of a cost function.

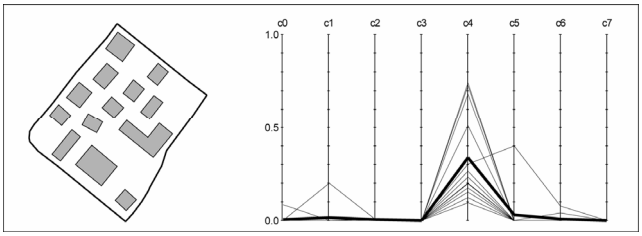


Fig. 3: Constraint violation after generalization

3.3 Cost Function

Automated generalization can be seen as an iterative process between conflict analysis and conflict resolution. Thus, the goal of generalization is to minimize the existing constraints violations as much as possible without creating new constraint violations. The difficulty stems from the fact that cartographic situations are connected to a set of constraints which partially work against each other. Examples are the constraint of ensuring minimal distances (MinDist; c2) between objects which works against the constraint that seeks to maintain positional accuracy (DiffPos; c4) or the need of reducing details (Lo-

calWidth; c3) versus the constraint of preserving the original shape (DiffEdgeCount; c5). The goal of automated generalization, then, is to find a good compromise between all these several constraints. Thus, a minimal equilibrium between all *n* constraints must be found. This can be expressed by a cost function that is, for example, just the simple average of all constraint values. In order to favor and punish certain constraints a weighting can be applied to them when calculating the cost function:

$$\textbf{Cost} = \sum \textbf{(Constraint * weight)} / \textbf{n}$$

The weights are scaled to [0, 1] and add up to 1. The constraints and their weights are the only parameters that can be adjusted in order to modify the system. For example the positional accuracy (DiffPos) could be weighted lesser or more depending on the map type. Especially the weighting of the soft constraints (the preserving constraints c4 to c7) can be used to tune and modify the selection of generalization operators. Weighting the soft constraints too low would result in a complete deletion of all buildings that are creating a conflict. In contrast high weights for the soft constraints would prohibit any changes as every change would only increase the cost further. In our experiments we have found that equal weights for all four soft constraints works well with our optimization approaches. Constraints are used to validate the result of generalization operators and thus to select the appropriate operator sequence. Obviously, this validation can only validate what is formalized in the constraints. The results of the optimization techniques presented here can always only achieve the quality that is formalized by the constraints. Thus, the correct and complete setting of the constraints is crucial for the correct validation of the result.

3.4 Effects of Generalization Operators

Typical generalization operators for our example of building generalization include simplification, exaggeration, aggregation, elimination, and displacement. These algorithms focus either on the removal or on the geometrical transformation of buildings. In automated generalization a particular generalization operator, e.g. building displacement, can be realized with different algorithms based on different solution approaches. Some operators have a narrowly defined functionality or are only applicable to specific feature classes, while others combine different functionalities. For instance, building typification, as realized in the algorithm proposed by [10] combines elimination, simplification, exaggeration, and displacement as it replaces a group of buildings by a placeholder. In this context the typification operator is comparable to an aggregation algorithm.

It is important to note that there are two types of generalization operators which are separated by a fundamental difference in terms of their granularity. The first group of operators, including aggregation, typification and simplification, is applied in discrete steps, usually removing entire features or details of features. These operators generate results that are absolute and irreversible. In contrast, continuous operators are only making slight and, more importantly, reversible changes. The most prominent example is the displacement operator which can redo positional changes in a later step. Likewise, the enlargement (exaggeration) and shrinking of features can be redone in a later processing step. This fundamentally different behavior of generalization operators is important when sequences of operators are created. An initial deletion of a map object can never be redone even if after several other generalization steps there would be sufficient space to retain the object. The displacement of features can, however, be reverted if the inverse displacement is applied in a later step.

Figure 4 shows the generalization operators that were used for the constraint optimization experiments. For every operator the result and the constraint violations are shown and can be compared with the "initial situation". Note that the sequence shown in this figure serves merely as an example to illustrate the various generalization operators. The values and behavior of these constraint violations may be very different for other building partitions. The implementation of the generalization operators shown here is described in more detail in § 5.2.

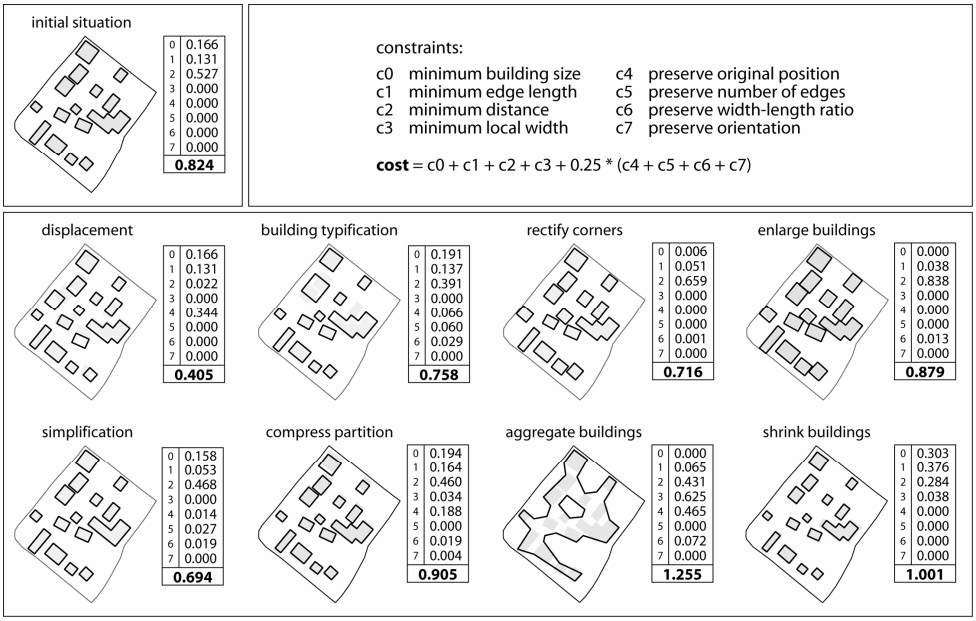


Fig. 4: Constraint violations for an initial map situation and after the execution of several generalization operators

It can be seen that the different operators are having quite different influences on the constraints. In the initial situation the minimum building size (c0), the minimum edge length (c1) and the minimum distance (c2) are violated. Operators for fulfilling the minimum size constraint are "enlarge buildings", sometimes "enlarge to rectangle" and under special conditions "aggregate buildings" and also "building typification" as they usually are removing the small buildings. For the minimum edge length constraint (c1) as well as for the minimum local width (c3) the "simplification" is the most obvious operator but also the before mentioned enlargement operators tend to improve the situation. "Displacement" reduces the violation of the minimum distance constraint (c2) but the original position can not be preserved (c4). If not enough space is available to displace all buildings sufficiently also other operators like "compress partition" and "shrink buildings" or operators that reduce the number of buildings such as "building typification" and "aggregate buildings" may help. These operators may, however, influence the preservation of the original position (c4), the number of edges (c5), the width-length ratio (c6) and the orientation (c7).

4 Processing Strategy – Methods

This paper proposes and evaluates combinatorial optimization strategies for the automated selection and chaining of generalization operators (application in the optimal sequence). The strategies are based on the capability to formalize goals and requirements as constraints in order to evaluate the results of different generalization operators. Operator sequences are created by iteratively executing all the available generalization operators and then selecting one or more results using a heuristic in order to continue the process, again executing all the operators. The search algorithms used in this paper are *hill climbing*, *simulated annealing* and *genetic search* (cf. § 4.2)

The generalization operators used in the processing strategies are completely independent from each other. Thus, they are behaving like a black box that receives input data with parameters and simply returns a result. Through the continuous evaluation of these results the processing system is basically self adapting. The operators are exchangeable and new operators can be added without having to make any changes.

4.1 Search Space

With a set of operators there exists a large space of possible operator sequences which can be pursued iteratively. The aim is to find an optimal result in this global space of possible solutions. Thus, global optimization addresses the task of finding the best operator sequences. These combinatorial problems

are NP-hard [12]. In the experiments presented below (see §6) we use eight operators and a maximum sequence length of 20 steps, resulting in a search space that has at most 1.31E+18 (derived from $\sum \#operators^{depth}$) possible solutions. Thus, for the illustrations in this section a smaller search space is used. Figure 5 shows such a search space of possible results for the processing with three operators and a maximum sequence length of three. At least one of the 40 possible states (initial state, 12 intermediate and 27 final states) has a minimal cost.]

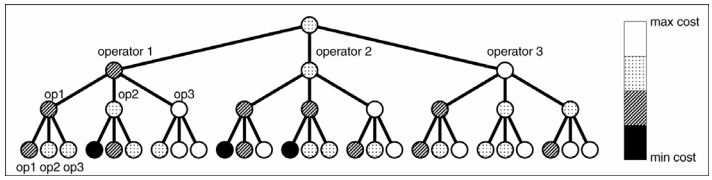


Fig. 5: Search space for combinatorial optimization with three operators

An example of the search space for an sample building block can be seen in Figure 6. Each line-up shows the costs of one of the 27 possible operator sequences with length three. At the root of the search space (depth 0) the costs are the same for all sequences. It can be seen that quite different sequences can lead to minimal costs (e.g. sequences 16 and 25) and even some others result in a near optimal result (e.g. sequences 4, 8, 10, 11, 13 and 22). But it can also be seen that the wrong sequence or operator selection can lead to quite a bad result (e.g. sequence 18) and that the optimal result in one sequence is not always achieved in the final step (e.g. sequence 6 and 18). The two lowest minima for this example (sequences 16 and 25) are both in quite irregular groups surrounded by rather bad results with high costs. The sequences 1 through 9 have all rather minimal costs as they all start with a displacement. Here the sequences 4 and 8 are two local minima designating the minimal costs for sequences starting with a displacement.

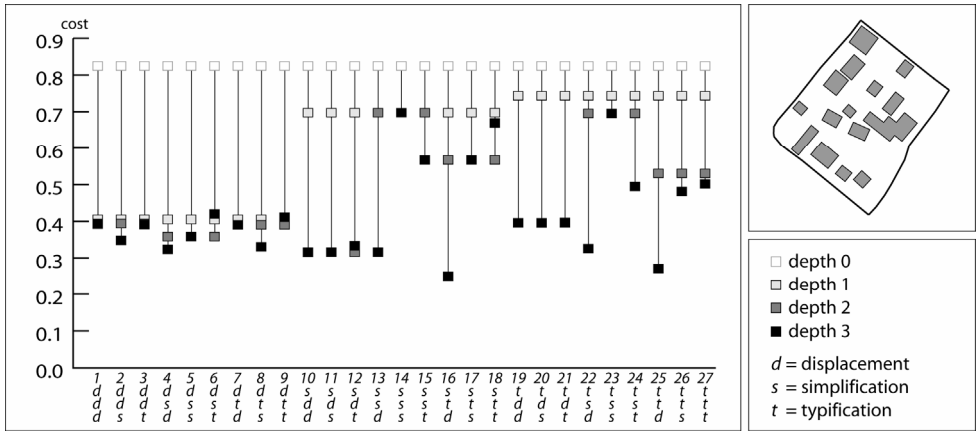


Fig. 6: Possible cost outcomes from a search space with three operators

Figure 7 shows the outcomes of three examples taken from the sequences of Figure 6. Sequence 4 shows a solution which is acceptable but there are still many small buildings which violate the minimum size constraint and some violations of the minimum distance constraint remain. Sequence 16 shows the best result which is achievable with only three operators. The number of buildings has been reduced slightly but there are still some minor violations of the minimum distance constraint. Sequence 18 proceeds in the first two steps similarly to sequence 16 but the final typification removed too many buildings, thus inducing excessive changes in the resulting map. In particular the constraints “original position” and “number of edges” (due to the removal of complete buildings) are violated severely. Thus, the final cost is much higher than the cost of the two other sample sequences.

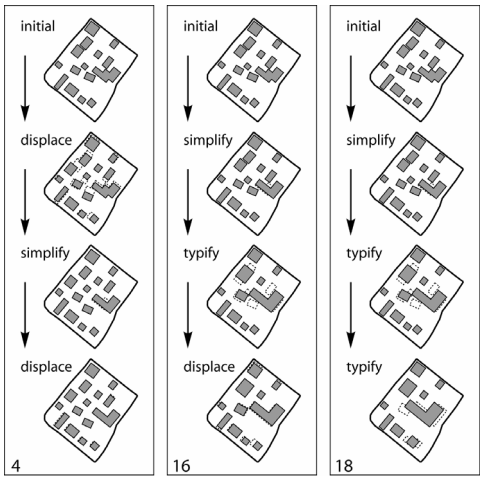


Fig. 7: Result examples for operator sequences from figure 6

It is important to note that the examples in Figures 6 and 7 might not be cartographically perfect solutions. This is due to the fact that only three generalization operators with limited capabilities were used. The examples are just used to illustrate the large diversity of results that are obtained when generalization operators are applied in permutated sequences.

In the search space example shown here three operators and a maximum sequence length of three was assumed. Thus, if all operators are used, every single one can only be used once. As shown in §3.4 there exist different types of operators. Some operators, including simplification, rectification or enlargement have to be executed only once. On the other hand, the typification operator may be used multiple times if the amount of buildings needs to be reduced substantially. Displacement in particular might be used at multiple points throughout a generalization process: In the beginning to remove distance problems, but also after a typification, aggregation, or enlargement of buildings in order to solve distance problems created by the preceding operators. Thus, the maximum sequence length should be longer than the number of available operators. In the experiments reported in § 6 a sequence length of 20 was used with 8 operators. It turned out that the average sequence length is between 6 and 11 depending on the search algorithm used.

4.2 Search Algorithms

As shown above, even a moderately large number of operators and moderately long maximum sequence length result in a huge global space of possible operator sequences. A brute force approach would test all the sequences, forming a tree of results (see Figure 5), in order to obtain the best result. This *deep exhaustive search* is definitely too slow as it would result in millions of executions of every generalization operator for only a single building partition. Thus, a heuristic is needed as a simplifying strategy in order to find a sequence with sufficiently minimal costs.

4.2.1 Hill Climbing

The most obvious strategy is the hill climbing approach. During the iterative optimization cycles always the currently best result is taken to proceed. Hill climbing approaches are proposed, for example, by [4]. This search algorithm assumes that at the beginning certain constraints are violated. The goal in the iterative processing cycle is to reduce these constraint violations as much as possible without violating others too much. Thus, only cost improvements are possible. A sequence which starts of with first increasing the cost in order to enhance the outcome of a following operator is not possible.

The basic processing cycle of the hill climbing algorithm is as follows:

(1) compute the constraint violations (value *cost before*) for the *current data*

(2) execute every available operator with a copy of the *current data*

(3) compute the constraint violations (value *cost after*) for every operator result

(4) if the *cost after* of the *best result* is lower than *cost before*: keep the *best result* as *current data* and the *cost after* as *cost before* and then continue with step 2
otherwise: return the *current data* as the final result of the processing cycle

This hill climbing approach is straightforward to implement and performs quite fast as only one sequence has to be computed per building partition. Figure 8 shows that the hill climbing approach always takes the best current result in order to proceed. Therefore it can get caught in a local minimum instead of finding the global minimum.

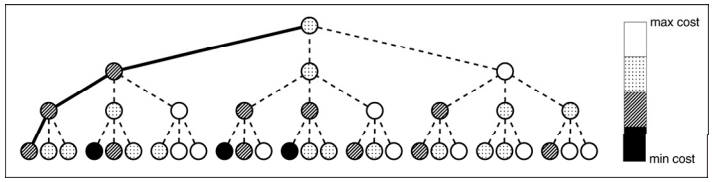


Fig. 8: Hill Climbing Algorithm, proceeds at every step with the best result (triplets ordered by cost from right to left)

In Figure 6, sequence 4 corresponds to the hill climbing approach. The result in this example is not perfect but still acceptable; this corresponds also with the experimental results over a larger number of building partitions reported in § 6.

4.2.2 Simulated Annealing

Simulated annealing (SA) approaches try to reduce the likelihood of getting caught in a local minimum. Instead of always taking the currently best result also an inferior result or even a result which increases the cost can be taken with a probability *P*. The probability *P* is continuously decreasing over time so that at the beginning of the generalization process the selection of results other than the best ones is more likely than towards the end. Thus, in the beginning SA chooses operators almost randomly, while in the end it resembles a hill climbing approach. The processing cycle is very similar to the hill climbing approach; only step 4 has to be changed:

(4) with probability *P*: retain another result than the *best result* as new *current data*, decrease *P* and then continue with step 2
with probability 1-*P*: if *cost after* is not lower than *cost before*: return finally the *current data*

The heavy dotted lines in Figure 9 show which sequence probably can get selected by this search algorithm. Thus for this specific example it is possible to get a better but also an inferior result.

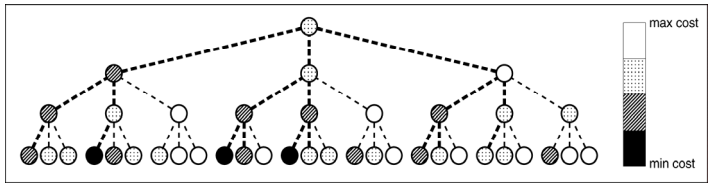


Fig. 9: Simulated Annealing, proceeds with another result than the best one with decreasing probability over time

The use of simulated annealing in map generalization has been shown by the example of a displacement algorithm [35] and a displacement algorithm in combination with other generalization algorithms [36]. In the approaches by Ware et al., however, every single building is treated separately in order to decrease the constraint violations. In contrast, the approach presented in this paper applies with every processing step an arbitrary stand-alone generalization operator onto an entire building partition. For example in a particular step all buildings of a partition may be subjected to a simplification operation.

4.2.3 Genetic Deep Search

In contrast to the two preceding heuristics, so called genetic algorithms do continue with a subset of the intermediate results instead of only one. [34] demonstrated the use of a genetic algorithm for the displacement of buildings. The genetic deep search strategy shown here starts off by retaining all operator results as intermediate results. With every step the number of results, that are retained to proceed, decreases, so that after some steps again only the best result is selected to proceed. This strategy searches not only one possible sequence but a set of sequences. As indicated in Figure 10 by the heavy lines, in the beginning all three results are retained and then at the next step only the two better results are cho-

sen to proceed. So from the 27 possible sequences in this example, six are evaluated instead of only one.

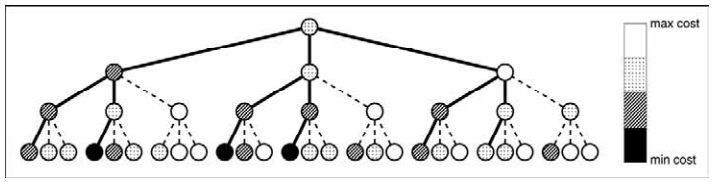


Fig. 10: Enhanced Deep Search, proceeds parallely with the best and a (decreasing) number of other results

In the experiments with eight operators (see §6) in the first iteration all eight results were retained, then four, two in the fourth and all following iterations only the best result was selected. This resulted in the evaluation of 64 sequences, each having a maximum length of 20. As with this approach 64 sequences instead of one have to be computed and evaluated this search algorithm is much slower than the two preceding approaches. However, as the experiments showed the likelihood of finding a better minimum is much larger.

5 Implementation as a Processing Service

The implementation was carried out in a web service framework called WebGen [7], utilizing operator services and support services (described in detail in [22]). The processing services control the sequencing of the different available operator services. Furthermore, they are also using functionalities offered by support services. Most of the currently available services in the WebGen framework are written in Java and thus can be used both as web services over a network or as Java classes on a local computer. In both cases the same interface is used. A client for the WebGen framework is available, among others, for the JUMP Unified Mapping platform [15]. Thus, JUMP is used for accessing and analyzing the processing services. Figure 11 shows the display of a parallel coordinate plot of constraints used for analyzing the results of a processing cycle within JUMP. Results of the various operator and processing services can directly be evaluated and visualized.

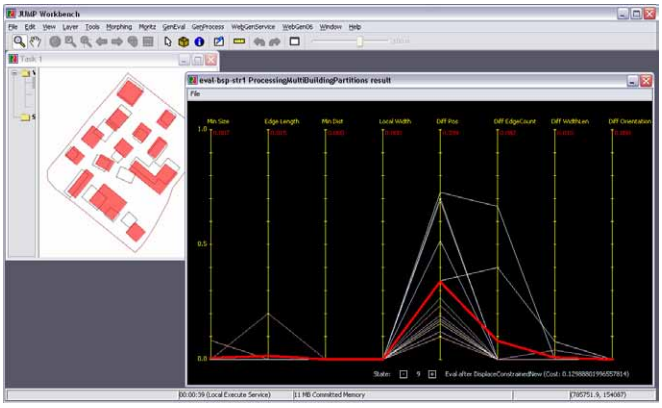


Fig. 11: Screenshot showing constraint violations after generalization within the WebGen framework

The implemented processing services first use a partitioning service to derive the individual building blocks. Then, for every partition (i.e. building block) the available operator services are called and their results evaluated by an evaluation support service. The search then proceeds over the following iterations: repeatedly calling the available operator services and the evaluation support service, according to the search algorithms outlined in the previous section.

5.1 Support Services

Two kinds of support services are used by the processing services, namely partitioning and evaluation services. The partitioning services are deriving building partitions based on the trans-hydro graph [33], utilizing the transportation and river networks for the subdivision of the map space into independent generalization zones. This partitioning strategy is sufficient for moderate scale changes as in our examples.

Evaluation services are applied for the calculation of constraint violation values for the building partitions before and after generalization. For every constraint a separate evaluation service is available which returns the violation costs. These values are then combined and evaluated by the cost function (see §3.3). All evaluation services have the same service interface. Thus new constraints can easily be added by creating a new evaluation service and adding it to the cost function.

5.2 Operator Services

The processing strategies presented in this paper have the goal to automatically chain all sorts of independent generalization operators. Every operator that is available as a service within the WebGen framework can be used by the processing system. They just have to be registered with the processing service. Thus, new operators can easily be added and due to the constraint based evaluation new operators are automatically integrated and applied in a sequence if it is appropriate. Operator services are developed to offer the generalization functionalities as for example simplification, displacement, typification, or scaling. In Table 1 the generalization operators used in our experiments are briefly described. Examples can be seen Figure 3.4. For downloads and more information on these and other algorithms, see the WebGen server (http://webgen.geo.uzh.ch). Note, however, that the generalization algorithms that have been implemented so far are mainly for demonstrations purposes of the web services and the processing services. Hence, they are simple algorithms taken from the literature. Designing optimally efficient and effective generalization algorithms was not the objective of our work.

Operator name	*Description*	*References*
Enlargement	Simply scales buildings that are smaller than the minimum building size about the building centroid.	
Simplification	Removes edges that are too short as well as small gaps between not directly connected edges.	[32]
Rectification	Simplification and rectified enlargement of buildings smaller than the minimum building size.	M. Bader in [1]
Displacement	Aims to achieve sufficient distances between the map features so that they are distinguishable and do not visually coalesce. The algorithm can respect different required minimum distances between buildings and between buildings and roads.	[22]; uses the triangulation-based proximity graph by [25]
Typification	Reduces the number of buildings and tries to maintain the arrangement of buildings in a partition. Uses a weighted Delaunay triangulation to replace nearby buildings with a placeholder resembling the more important replaced building.	[10]
Aggregation	Aggregates nearby buildings; growing and subsequent shrinking of buffers (dilation and erosion) are used to merge the buildings.	
Shrinking	Simply scales down the size of the buildings by a constant factor.	
Compression	Compresses and pushes all buildings away from a conflict zone, e.g. a road which is too close, using a rubber-sheeting like approach and reduces the building size accordingly.	

Table 1: Operator Services used in the experiments

5.3 Efficiency – Speed-up by Parallel Processing

The generalization algorithms and workflows shown so far are executed in a sequential, stepwise order. The presented processing strategies use a brute-force search approach by trying out all available algorithms on a particular partition of the map data. Especially for the genetic deep search approach huge amounts of processing steps have to be performed. Thus, instead of using an intensive, logically very complex search heuristic the more extensive search approaches try to solve the optimization tasks by trying out many different possible sequences. Parallel processing of separate partitions as well as the parallel execution of the algorithms on a single partition can improve the performance of these processing strategies substantially.

Parallel computing is no longer limited to specialized hardware and software platforms. The advent of multi-processor computers based on standard PC technology in combination with multi-threaded or distributed programming offers parallel computing functionalities to almost everybody. There exist a

great variety of parallel computing approaches. For the processing strategies presented in this paper a multiple instruction and multiple data stream (MIMD) system can be used. This approach subdivides the data and the instructions into independent tasks and computes them in parallel. The advantage is that no concurrency problems occur while accessing the data. This approach can be used both on a computer with multiple processors but also in a cluster of multiple computers linked via a network.

For the parallel processing of generalization tasks both domain and functional decomposition can be used [17]. Domain decomposition divides a job into independent small units. The job in our case consists of all the buildings that have to be generalized. The separation into independent partitions (i.e. the building blocks) delivers small units of independent data for generalization. Functional decomposition divides a workflow into different tasks as we do by applying the different algorithms onto a building partition in every iteration step. The parallel processing approach presented in this paper executes only completely independent tasks in parallel, which avoids message passing between the separate parallel processes.

Both domain and functional decomposition can be used in a service based environment using a cluster of multiple instances of the generalization services as shown in Figure 12. The *Parallel Processing Service* acts as a master process which derives the building partitions and creates a separate instance of the *Processing Service* (1) for every building partition $i = 1..n$. Every service instance is a completely separated process which receives its input data and returns its result data. In every instance of the *Processing Service* a processing strategy, such as the hill climbing search, is executed. During its processing cycle with every step all available *Operator Services* $j = 1..m$ are applied to the current map state of partition i (2). Thus, every *Operator Service* is applied to the same data. This can again be done in separate service instances. After all *Operator Services* have completed, their results are evaluated in order to retain one of the results. The computation and evaluation of the constraint violations can again be carried out in separate instances of these *Support Services* for the result each operator $j = 1..m$ (3). Thus at three stages of the processing strategies it is possible to create independent instances that can run in parallel.

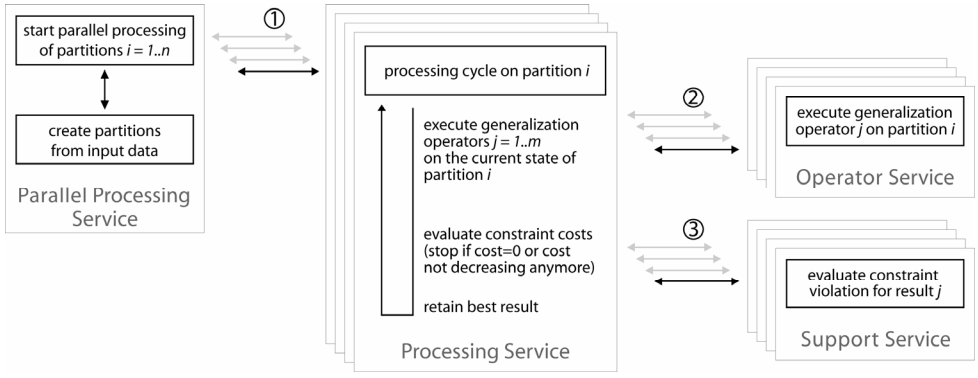


Fig. 12: Parallel processing in the WebGen framework

Every instance of a service is a process which runs completely independently. The different instances can run on a single computer or also on different computers, due to the web services architecture used. Thus, the load can be distributed. When using the processing services within JUMP on a local computer a separate thread is created for every service instance. These threads can run in parallel if the computer has more than one processor core as it is increasingly the case in standard personal computers. Thus, an immediate speed up is possible. Experiences and results with parallel processing on a multi-processor computer are presented in §6.4.

Different partitions can also be processed on different WebGen servers residing on different computers and also the instances of the operator services can be distributed on the different computers (i.e. nodes) of such a parallel processing cluster. A central controller executes in this case the *Parallel Processing Service*. This service calls the *Processing Services* on the different nodes. Currently we use a simple round robin scheduler for this purpose. However, a more advanced scheduler can be imagined which monitors the performance and current load of the various nodes and thus distributes jobs accordingly.

With the distributed parallel processing approach an overhead exists due to the duplication and distribution of the map data to the different services. However, as the execution time of the operators usually is

much longer than the network transfer time of the relatively small partitions, this overhead only a problem when only small numbers of building partitions are processed.

6 Experiments & Results

In this section experiments with the processing strategies and their different search algorithms are presented. The WebGen framework with the JUMP client served as the test platform.

6.1 Settings

For the experiments 99 sample partitions (building blocks) from the Swisstopo VECTOR25 dataset are used. Every test dataset consists of the buildings in a block and their surrounding roads. The type and density of the building partitions ranges from inner city building blocks with many, closely spaced buildings to rural partitions with only few, loosely distributed buildings. The number of buildings ranges from 1 to 80 per partition.

The cost function for the experiments uses the constraints introduced in § 3. The four constraints MinSize, EdgeLength, MinDist, LocalWidth were used as hard constraints with weight 1 since they absolutely should be fulfilled. The remaining four constraints (DiffPos, DiffEdgeCount, DiffWidthLen, and DiffOrientation) were understood as soft constraints for preserving certain properties; they were assigned a weight of 0.25. This value was chosen only for these experiments and proved to meet the needs. Using this weighting the cost values range from 5, if all constraints are maximally violated, to 0, if all constraints are perfectly satisfied. In our experiments the average initial cost was around 1.1 and the maximum initial cost around 2.5 with our sample data.

Before starting the processing cycle, four parameters ensuring the legibility of the resulting map are to initialize the constraints. All other parameters for the generalization operators are derived automatically from these four initial parameters. These basic parameters are the minimum size of a building, the minimum distance between buildings, the minimum distance between buildings and roads, and the minimum segment length of the building polygons. The parameter settings for the experiments are listed in Table 2.

For the experiments the derivation of a 1:50'000 map from the 1:25,000 VECTOR25 source data was performed. Therefore, we extracted our settings from the recommendations of the Swiss Society of Cartography [31]. At the 1:50,000 scale the complete traffic network should be maintained. All settlements must be shown as well as isolated individual buildings. Town centers should be maintained and in case of doubt houses can be omitted in favor of the traffic network. According to the legibility recommendations the smallest square on a 1:50,000 map should have an edge length of 0.35 mm on the map and thus of 17.5 m on the ground. This leads to a minimum building size of 306 m^2. The minimum distance between buildings on the map is 0.20 mm (i.e. 10 m on the ground), and the minimum segment length is 0.25 mm on the map (i.e. 12.5 m on the ground). A typical road in rural as well as residential areas has a ground size of 5 m. In the 1:50,000 map it is represented by a line with a thickness of 0.6 mm and thus has a virtual ground width of 30 m. Therefore the minimum distance between buildings and the road axis has to be 15 m.

minimum building size	306 m^2
minimum building distance	10 m
minimum building-road distance	15 m
minimum polygon segment length	12.5 m

Table 2: Processing parameters used for initializing the constraints

6.2 Comparison of Search Algorithms

In order to compare the different search algorithms the above settings were used. The different search strategies with their processing cycles were applied to each of the 99 building block samples. The most important value is the *average cost* as a measure of the constraint violations after the generalization of all samples (see § 3.3). Every strategy achieves a reduction of the average cost compared to the initial state (Table 3). The hill climbing approach (HC) and the simulated annealing (SA) reduced the average cost by 61% and 63%, respectively, with quite a similar outcome. The genetic deep search (GDS), however, reduced the average cost by almost 78%. This is due to the fact that the genetic approach tests at most 64 possible sequences instead of only one single sequence in the other search algorithms. This

can be seen by the *average sequence length* and the *calculated steps*. The two approaches HC and SA have an *average sequence length* as well as *average steps calculated* of 6.7 and 9, respectively, meaning that they calculate exactly the same number of steps (operator executions) for every sample. For GDS, however, the *average sequence length* is 11.6 for obtaining the optimal solution, with 467 steps calculated on average for the evaluation of 64 sequences. Unfortunately, this leads to a much longer processing time. In our test environment (see also § 6.4) the average execution time for every partition was more than 18 minutes with GDS, while HC and SA only needed 24 and 32 seconds, respectively.

	initial state	*hill climbing (HC)*	*simulated annealing (SA)*	*genetic deep search (GDS)*
average cost	1.0062	0.3899	0.3695	0.2218
- improvement (from initial)		61.25 %	63.28 %	77.96 %
- improvement (from HC)			5.24 %	43.12 %
average processing time		6.73 s	10.47 s	326.21 s
- slowdown (from HC)			55.57 %	4747.10 %
average steps calculated		6.7172	8.9596	467.3737
average sequence length		6.7172	8.9596	11.6364

Table 3: Comparison of different optimization methods tested with 99 building blocks

This comparison of the different search algorithms shows that the genetic deep search reduces the average cost significantly (more than 40% less) compared to the two other approaches. This gain in cartographic quality, however, can only be reached with a significantly slower execution. The quality gain of simulated annealing (SA) compared to hill climbing (HC) is only moderate. To achieve this cost reduction of 5%, the execution time grows by 55%, however. This is mainly due to the fact that the average sequence length with SA is 2.2 steps longer than with HC. An explanation for this effect is that a mistaken selection of a continuous operator (§ 3.4) in an early step of a sequence can be fixed in a later step. Thus, the selection of a sequence that would lead to a high local minimum can be compensated by the execution of other additional operators. The selection of a bad sequence starting with a discrete operator (e.g. aggregation or typification) can, however, not be redone in a later step, leading to the same result as HC. Towards the end of their execution SA and HC behave quite similarly; at that late stage SA can not better prevent from being trapped in local minimum than HC.

[36] showed a SA approach originally based on a displacement algorithm which is extended with functions to delete buildings or change their size. The optimization steps in their approach have a very fine granularity making only small changes like the displacement of a single building at a time. These small steps do not have instantly large effects if the SA algorithm initially chooses the wrong operation. Mistaken displacements of single buildings can even be redone later. In contrast, the optimization steps of the approach presented in this paper have a quite coarse granularity because of the fact that with every step the complete building partition is treated by the operator. These changes can be much more severe and not reversible at later execution steps. The quasi-random operator choice in the beginning of a SA generalization process can then lead to a completely mistaken operator sequence. Thus, applying SA in this setting offers no substantial improvements over HC while having a less good performance.

6.3 Comparison of Operator Sequences

In order to better understand the behavior of the different search algorithms the positions where the particular operators are executed at in a processing sequence can be analyzed. Figure 13 compares this for hill climbing (HC) against simulated annealing (SA). Let's first look at the circle sizes only, without distinguishing between HC and SA. Apparently the displacement operator is used much more often than most of the other operators, as indicated by its large circle size. Usually this behavior is caused by minimum distance problems, particularly between buildings and roads. The enlargement operator is executed quite rarely, usually at most once per sample. The shrinking and compression operators both are more likely to get executed later in a sequence because they can sometimes resolve conflicts that are not solvable by displacement. As shown in the examples of Figure 4 the typification, aggregation, shrinking and compression operators can not always reduce the cost by themselves but they are needed to reduce the number and size of buildings in order to render the displacement operator successful. Discrete operators such as simplification and rectification are executed relatively often as they usually are required in every partition and they usually reduce cost without creating new conflicts.

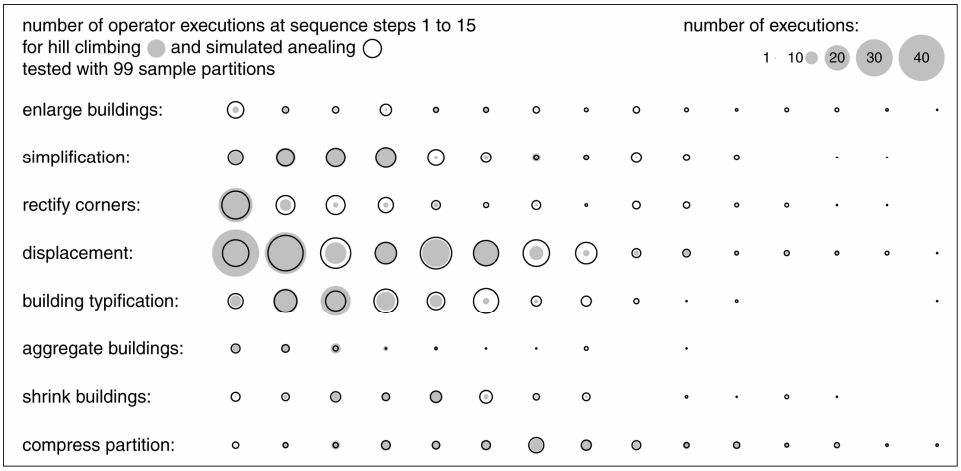


Fig. 13: Comparison of hill climbing and simulated annealing operator sequence positions

Comparing HC and SA it can be observed that the overall number of operator executions does not differ much. The distribution of operator selections in the beginning, however, is more uniform for the SA approach as all operators have a chance of getting executed due to the initially randomized behavior. HC, however, favors the operators which promise a high cost reduction, thus especially the displacement operator and more seldom rectification, simplification and typification. The other operators get executed very rarely in the beginning of a sequence. If SA is used, displacement is not executed as often in the beginning as with HC; however, it is selected somewhat more often in later steps than with HC. This behavior applies also for the typification. Apart from these observations the two strategies behave very similarly.

The comparison of the operator positions for the hill climbing (HC) and the genetic deep search (GDS) approaches is shown in Figure 14. The first striking difference is that GDS uses many more executions than HC. This is clearly true as evidenced in Table 3 by the average sequence length. GDS has an average length of 11.6 against 6.7 for HC. Thus, almost double the number of operator executions take place in an average sequence.

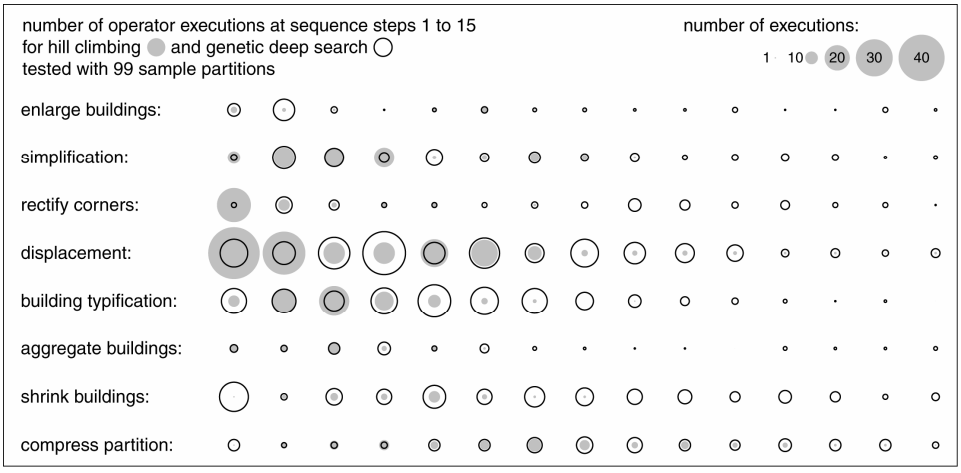


Fig. 14: Comparison of hill climbing and genetic deep search operator sequence positions

When comparing the two approaches HC and GDS it can be seen that especially the shrinking but also the typification operators are used much more often with GDS. With HC these operators are not very favorable because they are initially not reducing the cost very much. GDS, however, evaluates also these sequences that start off with such a less favorable operator. Thus a sequence can be found that leads to a better global minimum than with HC.

6.4 Parallel Processing Performance

As already introduced in § 5.3 a parallel processing approach was used in order to speed up the process. In a two stage approach during the iterative processing cycle for every partition the executions and evaluations of the different operators were computed in separate, parallel threads. The execution of these independent processing cycles again was parallelized so that more than one partition was treated in parallel. This strategy, outlined in Figure 12, helped to overcome the problem that every iteration cycle is only as fast as the slowest operator as the evaluation and selection can only be carried out when all results are available. The experiments were conducted on a server equipped with four dual-core processors running at 2.19 GHz clock speed under the Windows 2003 Server operating system. Figure 15 shows the CPU usage history for the processing of a dataset with 9 building partitions using the hill climbing approach. With sequential execution the first two processors (with one CPU used for the scheduler) get used, but as the program is sequential, the system load never exceeds the 13 % of a single processor. With parallel execution all eight CPUs get used and thus the complete system load is approximately 60 % for this example. The complete processing time decreases from 33 s (sequential) to only 17 s (parallel).

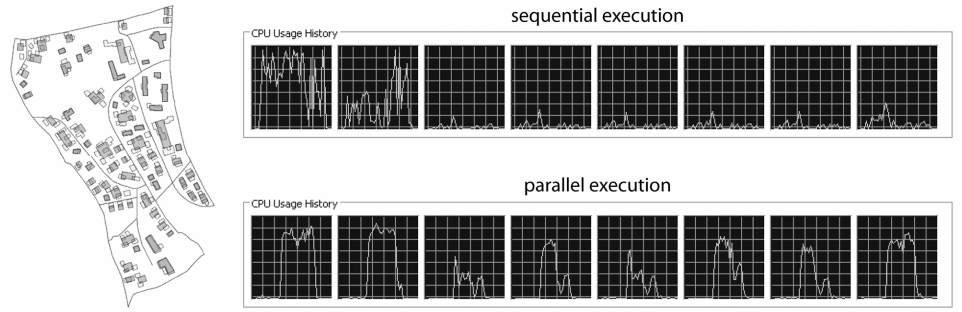


Fig. 15: Parallel processing CPU usage: 33 seconds (sequential), 17 seconds (parallel)

Thus, already with nine sample partitions a significant speed up could be achieved. The processing time has been reduced to 51 % compared to the sequential approach. The system load fluctuates due to interferences between the Windows scheduling and the thread scheduling of the Java Virtual Machine. Working with larger numbers of sample partitions the performance gain can be increased to a constantly higher system load than the 60 % in the example above. In performance tests with all 99 sample partitions the system load reached the 100 % particularly during the initial phase when still many small partitions were processed in parallel and only later dropped to a low of 13-20 % towards the end when only few large and complex partitions remained to be finished. An intelligent scheduler knowing about this problem could increase the performance by trying to mix the processing of complex and simple partitions. The processing of 99 samples (same as used for Table 3) with the hill climbing approach took 2,799 seconds with sequential and 667 seconds with parallel execution, yielding a reduction by 76 %. The average overall system load was approximately 51 %.

7 Discussion

7.1 Global vs. Local Constraint Satisfaction

The processing strategies used in this paper are all based on the ability to describe the desired cartographical quality sufficiently by means of constraints. Currently the constraints are only evaluated globally for each partition, that is, the approach is controlled by the satisfaction of the constraints of all features in one partition (i.e. the cost function in §3.3). Remedies for specific constraint violations of one single map feature are therefore in the responsibility of the individual operators as they are usually trying to solve the most important constraint violations first.

7.2 Approximating Deep Search

Among the three tested heuristics the *genetic deep search* approach provided the reference results. This approach tries to approximate a complete exhaustive search and thus to test as many as possible different sequences. A real exhaustive search is clearly not feasible due to the overwhelming complexity. Another combination of a search heuristic and a *deep search* could be the concatenated use of deep

searching with low depth. For example, compute the full search space with depth 3, then take the best result and proceed again by computing this limited search space for this intermediate best result. Such approaches will probably benefit significantly from the parallelization.

7.3 Combining Automated Processing with Pre-defined Workflows

The approaches presented in this paper only looked at the generalization of buildings constrained by their surrounding roads. In a real scenario these roads as well as all other map features must be generalized accordingly. Following the recommendations of [31] for the scale transition from 1:25,000 to 1:50,000 no real road generalization is necessary as most of the roads need not change and only few are eliminated (e.g. dead ends). For a larger scale change (e.g. 1:25,000 to 1:100,000) more substantial road generalization including elimination, simplification and displacement must be applied. Therefore a combined semi-automatic procedure using predefined workflows in combination with constraint based processing strategies could be used. A workflow modeling system [23] could then trigger different static and automated generalization services:

- road elimination by road-class (e.g. constrained by a connectivity graph)
- partitioning (buildings inside a block are selected, preserves topology)
- road simplification (including displacement if possible)
- building generalization of the partitions (constrained by the pre-generalized roads)
- road displacement (if needed and possible)

The two last steps possibly could be iterated in order to resolve major conflicts between roads and buildings. The workflows themselves can again be seen as one generalization operator. Such an opaque service would hide the complex workflow behind a simple interface. Even the use of such an operator in another workflow or automated process is imaginable.

7.4 Optimization through Machine Learning and Collaborative Filtering

Features or groups of map features can be placed inside a constraint space [8] depending on their cartographic properties. The distance of the features from the origin of this constraint space is a measure of cartographic conflicts. Features at the origin are not violating any cartographic constraints. The constraints allow the identification of similar cartographic situations for the features and partitions, which will be generalized with the same generalization operator. Thus, for example a situation with many minimum distance and minimum building size violations would indicate the use of a typification first, in order to reduce the number of buildings, followed by a displacement.

The method suggested by [9] uses the constraint space for the prediction of generalization operator sequences. This method exploits a knowledge base to predict the operator sequences and proceeds in two phases, a training phase and an application phase. In the training phase, the knowledge base is populated with information about successfully generalized features (described by their position in the constraint space) and the corresponding operator sequences and parameter settings that lead to the successful result. In the application phase the constraint patterns of a given map object are matched against the trained constraint patterns to retrieve similar operator sequences. Thus, the most promising operators with their associated parameter settings are applied to the map object. A collaborative filtering technique [18] guaranties a fast retrieval of operator sequences from a large knowledge base.

As the results reported in § 6.2 show, the *genetic deep search* approach may be prohibitively slow when the execution time plays a role. However, as it delivers significantly better results than the other approaches it could serve as a search strategy for the training phase of the knowledge base leading to the storage of better sequences than can be found with other search algorithms, such as hill climbing and simulated annealing.

8 Conclusion

Constraint based combinatorial optimization techniques can be used for controlling the selection and chaining of different generalization operators of varying sophistication and task granularity. The operators as well as the supporting functions and the iterative processing methods are implemented within

the WebGen generalization web services environment which was originally developed with the aim of building a common research platform [7] and later extended by additional functionality [22]. The operators used are true stand-alone services and have no direct interdependences; the web service interfaces are the only communication and data transfer utility. In this paper, we have now demonstrated how the versatility of a service based architecture can be exploited in generating automated generalization processes, in a way that is not possible with more traditional computing paradigms. In particular, the service based approach allows to build modular generalization processes that can be flexibly extended by introducing additional generalization operators or supporting facilities, that can be coupled to arbitrary optimization strategies, and that can be parallelized in order to speed up computationally heavy processing.

Different optimization algorithms including *hill climbing*, *simulated annealing* and *genetic deep search* can be used and produce different results in terms of amount of conflict reduction and processing performance. For the experiments partitions of buildings separated by a trans-hydro-graph were generalized automatically. The *genetic deep search* evaluates a range of possible operator sequences not only one like the other optimization algorithms. Thus, the amount of conflict reduction is significantly higher but the processing time needed inhibits use with larger datasets or in near real-time environments. It is interesting to note that with the presented processing strategy the *simulated annealing* approach only optimizes slightly better than the *hill climbing* approach while needing significantly more processing time. This finding is in contrast with the results of [36]; the main reason for this difference is the different operator granularity (see §6.2). Parallel processing proved to be useful for increasing the process performance. As a future step the combination of the *genetic deep search* approach with machine learning techniques will be investigated. Initial work in this direction has been reported in [9].

Acknowledgements

This research was partially funded by the Swiss National Science Foundation (grant 20-101798, project DEGEN). Thanks go to Ingo Petzold and Stefan Steiniger for helpful comments.

References

[1] AGENT Consortium. “Deliverable D1 – Specification of basic algorithms”. Department of Geography, University of Zurich, 1999.

[2] M. Bader, M. Barrault and R. Weibel. “Building Displacement over a Ductile Truss”, International Journal of Geographical Information Science, Vol. 19, No. 8-9, pp. 915-936, 2005.

[3] S. Bard. “Quality assessment of cartographic generalization”, Transactions in GIS, Vol. 8, No. 1, pp. 63-81, 2004.

[4] M. Barrault, N. Regnauld, C. Duchêne, K. Haire, C. Baeijs, Y. Demazeau, P. Hardy, W. Mackaness, A. Ruas, and R. Weibel. “Integrating multi-agent, object-oriented and algorithmic techniques for improved automated map generalization”, in Proc. 20th Int. Cartographic Conf., Beijing, China, 2001, pp. 2110-2116.

[5] K. Beard. “Constraints on rule formation”, in B. Buttenfield and R. McMaster (Eds.), Map Generalization: Making Rules for Knowledge Representation, London: Longman, 1991, pp. 121–135.

[6] D. Burghardt and S. Meier. “Cartographic Displacement Using the Snakes Concept”, in Foerstner, W., and L. Pluemer (Eds.), Semantic Modeling for the Acquisition of Topografic Information from Images and Maps, Basel: Birkhaeuser, 1997, pp. 59-71.

[7] D. Burghardt, M. Neun and R. Weibel. “Generalization Services on the Web – A Classification and an Initial Prototype Implementation”, Cartography and Geographic Information Science, Vol. 32, No. 4, pp. 257-268, 2005.

[8] D. Burghardt and S. Steiniger. “Usage of Principal Component Analysis in the Process of Automated Generalisation”, in Proc. of 22nd Int. Cartographic Conf.. La Coruña, Spain, 2005.

[9] D. Burghardt and M. Neun. “Automated sequencing of generalisation services based on collaborative filtering”, in M. Raubal, H. J. Miller, A. U. Frank, & M. Goodchild (Eds.), Geographic Information Science, 4th Int. Conf. on Geographical Information Science (GIScience), IfGIprints 28, 2006, pp. 41–46.

[10] D. Burghardt and A. Cecconi. “Mesh Simplification for Building Typification”. International Journal of Geographical Information Science, Vol. 21, No. 3, pp. 283-298, 2007

[11] B. Buttenfield and R. McMaster. Map generalization: making rules for knowledge representation. Longman: London, 1991.

[12] P. Gray, L. Painton, C. Phillips, M. Trahan, J. Wagner. “A Survey of Global Optimization Methods”, Technical Report, http://www.cs.sandia.gov/opt/survey/main.html (accessed 02/2007), 1997.

[13] L.Harrie. “The Constraint Method for Solving Spatial Conflicts in Cartographic Generalization”, Cartography and Geographic Information Science, Vol. 26, No. 1, pp. 55-69, 2000.

[14] L. Harrie and R. Weibel. “Modelling the Overall Process of Generalisation”, in A. Ruas, W.A. Mackaness and T. Kilpeläinen (Eds.), Generalisation of Geographic Information: Cartographic Modelling and Applications. Amsterdam: Elsevier Science, 2007, pp 67-87.

[15] JUMP. “The JUMP Unified Mapping Platform”, http://www.jump-project.org, 2007.

[16] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi. “Optimization by Simulated Annealing”, Science, Vol. 220, No. 4598, pp. 671-680, 1983.

[17] G. Langran. “Generalization and parallel computation”, in B. Buttenfield and R. McMaster (Eds.), Map Generalization: Making Rules for Knowledge Representation, London: Longman, 1991, pp. 204-216.

[18] G. Linden, B. Smith and J. York. “Amazon.com Recommendations. Item-to-Item Collaborative Filtering”. IEEE Internet Computing, Vol. 7, pp. 76-80, 2003.

[19] R. McMaster and S. Shea. Generalization in Digital Cartography. Washington, USA: Association of American Geographers, 1992.

[20] S. Mustière. “Cartographic generalization of roads in a local and adaptive approach: A knowledge acquisition problem”, International Journal of Geographical Information Science, Vol. 19, No. 8-9, pp. 937-955, 2005.

[21] S. Mustière, J.-D. Zucker, L. Saitta. “An Abstraction-Based Machine Learning approach to Cartographic Generalization”, in 9th Int. Symp. on Spatial Data Handling (SDH 2000), Beijing, China, 2000, pp. 50-63.

[22] M. Neun, D. Burghardt and R. Weibel. “Web Service Approaches for Providing Enriched Data Structures to Generalisation Operators”. Accepted for International Journal of Geographic Information Science.

[23] I. Petzold, D. Burghardt and M. Bobzien. “Workflow Management and Generalisation Services”, in 9th ICA Workshop on Generalization and Multiple Representation, Portland, 2006.

[24] N. Regnauld. “Constraint based mechanism to achieve automatic generalization using agent model”, in Proc. GIS Research UK (GISRUK 2001), University of Glamorgan, 2001, pp. 329–332.

[25] N. Regnauld. “Spatial Structures to Support Automatic Generalisation”, in Proc. of the XXII Int. Cartographic Conference, A Coruña, Spain, 2005.

[26] D. Richardson and J.-C. Muller. “Rule selection for small-scale map generalization”, in B. Buttenfield and R. McMaster (Eds.), Map Generalization: Making Rules for Knowledge Representation, London: Longman, 1991, pp. 136-149.

[27] A. Ruas and C. Plazanet. “Strategies for Automated Generalization”, in Proc. of the 7th Int. Symposium on Spatial Data Handling (SDH 1996), Delft, the Netherlands, 1996, pp. 319-336.

[28] A. Ruas. “Modèle de généralisation de données géographiques à base de contraintes et d’autonomie”, Ph.D. thesis, IGN France and Université de Marne La Vallée, 1999.

[29] A. Ruas and C. Duchêne. “A Prototype Generalisation System Based on the Multi-Agent System Paradigm”, in A. Ruas, W.A. Mackaness and T. Kilpeläinen (Eds.), Generalisation of Geographic Information: Cartographic Modelling and Applications. Amsterdam: Elsevier Science, 2007, pp. 269-284.

[30] M. Sester. “Generalization Based on Least-squares Adjustment”, International Archives of Photogrammetry and Remote Sensing. Vol. XXXIII, Part B4, Amsterdam, 931-938, 2000.

[31] Swiss Society of Cartography. “Topographic Maps - Map Graphics and Generalisation”, Wabern, Switzerland: Swiss Society of Cartography, 1995.

[32] W. Staufenbiel. “Zur Automation der Generalisierung topographischer Karten mit besonderer Berücksichtigung großmaßstäbiger Gebäudedarstellungen”, Institute of Cartography and Geoinformatics, University of Hannover, Nr. 51, 1973.

[33] S. Timpf. “Hierarchical structures in map series”, PhD Thesis, Technical University Vienna, 1998.

[34] I.D. Wilson, J.M. Ware and J.A. Ware. “A genetic algorithm approach to cartographic map generalization”, Computers in Industry. Vol. 52, No. 3, pp. 291-304, 2003.

[35] J.M. Ware, and C.B. Jones. “Conflict Reduction in Map Generalization Using Iterative Improvement”, GeoInformatica, Vol. 2, No. 4, pp. 383-407, 1998.

[36] J.M. Ware, C.B. Jones and N. Thomas. “Automated Map Generalization with Multiple Operators: A Simulated Annealing Approach”. International Journal of Geographical Information Science, Vol. 17, No. 8, pp. 743-769, 2003.

[37] R. Weibel, S. Keller and T. Reichenbacher. “Overcoming the knowledge acquisition bottleneck in map generalization: the role of interactive systems and computational intelligence”, in Proc. of 2nd Int. Conf. on Spatial Information Theory (COSIT 95), 1995, pp. 139–156.

[38] R. Weibel and G. Dutton. “Constraint-based automated map generalization”, in Proc. 8th Int. Symposium on Spatial Data Handling, 1998, pp. 214–224.

Research Paper 5

Burghardt D. and M. Neun (2006): Automated sequencing of generalisation services based on collaborative filtering. In: M. Raubal, H. J. Miller, A. U. Frank and M. Goodchild (eds): *Geographic Information Science*. 4th Inernational. Conference on Geographical Information Science (GIScience 2006), IfGIprints 28, pp. 41-46.

Automated sequencing of generalisation services based on collaborative filtering

Dirk Burghardt, Moritz Neun

GIS Division / Department of Geography / University of Zurich
{burg,neun}@geo.unizh.ch

INTRODUCTION

The need for automated control of generalisation operators is addressed by various researchers, for an overview see Jones and Ware (2005). Available approaches reach from simple batch processing, over condition-action modelling until sophisticated constraint based methods, which are compared and discussed in Harrie and Weibel (2006). The constrained based methods can be further subdivided in complex methods, which perform different operations simultaneously or the application of constraints to chain specific algorithms that perform one operation at a time (Mustiere 2005). Examples of the first category are for instance least square adjustment (Harrie 2000, Sester 2000), energy minimisation (Burghardt 1997, Bader 2001) or simulated annealing (Ware et al. 2003), whereas the AGENT approach (Ruas 1999, Regnauld 2001) belongs to the second one.

The presented approach falls also in the last category of constraint based approaches applied to the chaining of different generalisation operators. The combination of operators is situation dependent. The situation is given through the characteristics of the map features and their relations described by the cartographic constraints. Constraints can work partially against each other. For example the constraint of "preserving minimal distances" between features work against the constraint of "keeping positional accuracy" or the need of "reducing details" has a negative effect on the constraint of "keeping the original shape" as best as possible. The goal of automated generalisation is to find combination of operators, that a compromise between all these several constraints can be reached. The generalisation process is tested on the example of building generalisation, where several cartographic constraints can be formulated and different generalisation operators can be applied.

Service framework. The implementation is carried out in a service framework (Burghardt et al. 2005) with an utilisation of evaluation services, operator services and support services (Neun et al. 2006). *Suppport services* are used for the derivation of building partitions based on a trans-hydro

graph (Timpf 1998), whereby transportation and river network are utilised for the subdivision of independent generalisation zones. *Operator services* are developed to offer the generalisation functionalities as for example simplification, displacement, typification or scaling. Finally the *evaluation services* are applied for the calculation of constraint values for the building partitions before and after generalisation.

COLLABORATIVE FILTERING APPLIED TO AUTOMATED GENERALISATION

Collaborative filtering can be used for making automated predictions of user or entities preferences. Major applications are for example recommendations in e-commerce Web site (Linden et al. 2003) or Peer-to-Peer file-sharing systems (Wang et al. 2006). The focus there is on filtering relevant information in a personalised respective adapted way from a large amount of data.

Our approach applies this method to automated generalisation, for the prediction of generalisation operator sequences. A knowledge base is exploited to predict the operator sequences. The knowledge base stores information about successfully generalised features and their corresponding operator sequences respective parameter settings. The approach works for single features as well as for group of features. Therefore in the following the term *entity* is used. In the example below every entity represents a partition of buildings, derived automatically from a street and river network.

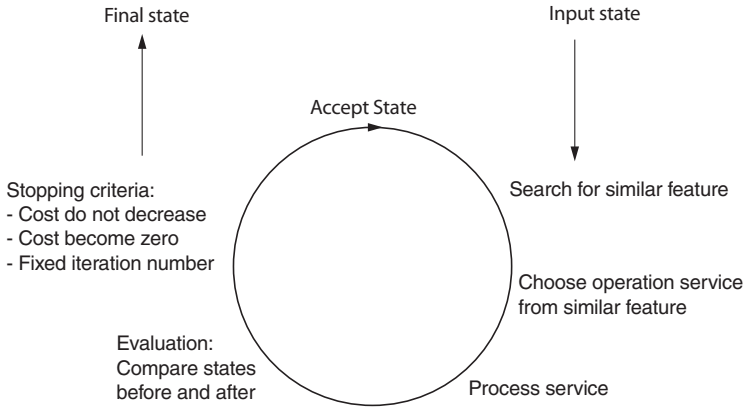


Fig. 1: Flow chart of the applied generalisation process

The generalisation process (Fig. 1) starts with the search for entries in the knowledge base, which have similar constraint patterns. Then the most

promising operators with their parameter settings are applied onto the entity. The collaborative filtering technique guaranties a fast extraction of operator sequences from a large knowledge base.

The prediction quality for the best suited operator sequences, depend on the number of entities in the knowledge base. At the beginning all generalisation operators, maybe with varying parameters are tested at every iteration step to find the best operator for the current situation. Values for the constraints and the evaluation of the generalisation operators are added to the knowledge base at every generalisation step. After a while the prediction quality improves and only the highest ranked operators have to be applied. The approach is implemented and results will be presented.

The knowledge base will be build up in a continuous way during an operative application of automated generalisation. At the beginning the knowledge base contains only one entity, which recommends all available generalisation operators equally (value = 1.0). Before and after application of the generalisation operators an evaluation based on constraints is carried out to derive an evaluation of the applied generalisation operators. The knowledge base gets added now a second entity consisting of constraint values (`EntityConstraints`) characterising the situation and evaluation values (`EntityOperation`) for the applied generalisation operators. The evaluation of generalisation operators is derived from the comparison of constraint values before and after generalisation.

Table 1 shows a small extract (4 entity states) of a knowledge base from building generalisation with constraint values characterising the situations (`EntityConstraints`) and the corresponding operation values for the evaluation of the applied generalisation operators (`EntityOperation`). The evaluation is based on 8 constraints (size, length, distance, length-width ratio, position change, edge number change, length-with ratio change and orientation change). There are 5 different generalisation operators applied as relative scaling, simplification, replacement of a building through rectangle, displacement and typification, the last one with two different parameter settings.

Tab. 1: Schema of an automatically created knowledge base for building generalisation with constraint values characterising the situations (`EntityConstraints`) and the corresponding operation values for the evaluation of the applied generalisation operators (`EntityOperation`).

EntityConstraints EntityOperation	Size Scal	Leng Simp	Dist Rect	LWid Disp	DPos Ty10	DEdg Ty30	DWLR	DOrie
EntityConstraints	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
EntityOperation	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000		
EntityConstraints	0,2169	0,6687	0,6946	0,5097	0,0000	0,0000	0,0000	0,0000
EntityOperation	1,9858	1,5035	2,0111	1,4354	2,0409	2,2188		
EntityConstraints	0,2169	0,6687	0,0009	0,5097	0,0392	0,0000	0,0000	0,0000
EntityOperation	1,3685	1,0148	1,3948	1,4345	1,4632	1,3988		
EntityConstraints	0,2169	0,3852	0,1969	0,0984	0,0354	0,0639	0,0172	0,0009
EntityOperation	0,9225	1,0148	0,9195	0,8268	0,8306	1,1236		

The generalised entity of the most successful operator will replace the ungeneralised one and can be used for further generalisation iterations (Fig. 1). If no more improvements based on constraint evaluation can be reached by the applied generalisation operators the iteration stops.

Fig. 2: Results of building generalisation (original – left, results – right) after automated sequencing of generalisation services. Data: VECTOR 25, reproduced by permission of swisstopo (BA057008).

REFERENCES

Bader, M. (2001). Energy Minimization Methods for Feature Displacement in Map Generalization, Doctoral Thesis, Department of Geography, University of Zurich, Switzerland.

Burghardt, D., M. Neun and R. Weibel (2005). Generalization Services on the Web – A Classification and an Initial Prototype Implementation. Cartography and Geographic Information Science, Vol. 32, No. 4, pp. 257-268.

Burghardt, D. and S. Meier (1997). Cartographic Displacement Using the Snakes Concept. In Foerstner, W., and L. Pluemer (eds), Semantic Modeling for the Acquisition of Topografic Information from Images and Maps, Birkhaeuser Verlag, pp. 59-71.

Harrie, L. (2000). The Constraint Method for Solving Spatial Conflicts in Cartographic Generalization. Cartography and Geographic Information Science, Vol. 26, No. 1, pp. 55-69.

Harrie, L. and R. Weibel (2006). Modelling the Overall Process of Generalisation. In: Ruas, A., Mackaness, W.A. and Kilpeläinen, T. (eds.). Challenges in the Portrayal of Geographic Information: Issues of Generalisation and Multi Scale Representation. Elsevier Science.

Jones, C.B. and J.M. Ware (2005). Map generalization in the Web age. International Journal of Geographical Information Science. Vol. 19, No. 8–9, pp. 859–870.

Linden, G., B. Smith and J. York (2003). Amazon.com Recommendations. Item-to-Item Collaborative Filtering. IEEE Internet Computing, pp. 76-80.

Mustiere, S. (2005). Cartographic generalization of roads in a local and adaptive approach: A knowledge acquistion problem. International Journal of Geographical Information Science. Vol. 19, No. 8–9, pp. 937–955.

Neun, M., D. Burghardt and R. Weibel (2006). Spatial Structures as Generalisation Support Services. Joint ISPRS Workshop on Multiple Representation and Interoperability of Spatial Data, Hannover, 2006.

Regnauld, N. (2001). Constraint based mechanism to achieve automatic generalization using agent model. In Proceedings GIS Research UK GISRUK 2001, University of Glamorgan, pp. 329–332.

Ruas, A. (1999). Modèle de généralisation de données géographiques à base de constraints et d'autonomie. Ph.D. thesis, IGN France and Université de Marne La Vallée.

Sester, M. (2000). Generalization Based on Least-squares Adjustment. International Archives of Photogrammetry and Remote Sensing, Vol. XXXIII, Part B4, Amsterdam, pp. 931-938.

Timpf, S. (1998). Hierarchical structures in map series. PhD Thesis, Technical University Vi-enna, Vienna.

Wang, J., J. Pouwelse, R. Lagendijk and M.R.J. Reinders (2006). Distributed Collaborative Filtering for Peer-to-Peer File Sharing Systems. Proceedings of the 21st Annual ACM Symposium on Applied Computing.

Ware, J.M., C.B. Jones and N. Thomas (2003). Automated Map Generalization with Multiple Operators: A Simulated Annealing Approach. International Journal of Geographical Information Science, Vol. 17, No. 8, pp. 743-769.

Part III

Appendices

Appendix A

The WebGen Framework

The WebGen Framework can be split up into two major parts. The first is the web services level with the communication and interface issues. The second part is made up of server and client components, which emerged from the WebGen prototype.

Service Communication and Interface Description

In an open research model every researcher may present his/her own generalization service. Through the Internet and the use of platform independent technologies such services may reside on servers all over the world. In order to make it possible these services the Registry (cf. figure A.1), some sort of "Yellow Pages", indicates which services are available, where they are located and what algorithms they offer (Burghardt et al., 2005b). The registry is a single access point where all further information is to be found. Whilst services can change or move they are always to be found again through the registry. This model for sharing and discovering generalization services can be summarized by the "publish-find-bind" paradigm (e.g. UDDI by OASIS, 2002) where the service is published, for instance, by its author in the registry. Once it is published in the registry the service can then easily be found and used (i.e. bound) by others.

The registry of the WebGen Framework is implemented as a dynamic web application using the scripting language PHP (Hypertext Pre-processor) and the MySQL database for storing the service entries. For every service a name, a description, the owner, the version number and the address of the service description is stored in this database. The client components of the framework can query this registry for available services and get the details as an XML document. With the supplied address the service's interface description, which is also an XML document, can be retrieved. This document can reside on any web server; usually it is either also on the registry server or on the server that offers the service. This interface description contains all the necessary information about the service, including e.g. the URL (Uniform Resource Locator) for the execute request and information about the input and output parameters of the service. The client components of the WebGen framework can use this information for generating the dialog for the service dynamically. Thus, the client components are generic and must not be updated every time a new service

is made available. The excerpt in listing A.1 shows an interface description for a buffering service getting a collection of map features with geometries and a double value for the buffer width as input, delivering as output a new collection of map features with the buffered geometries.

Listing A.1: WebGen interface description example

```
<?xml version="1.0" encoding="UTF-8" ?>
<webgen:Interface xmlns:webgen="http://www.webgen.org/webgen">
   <webgen:name>SimpleBuffer</webgen:name>
   ... further: author, service category, endpoint,
       service name, version and description ...
   <webgen:InputParameters>
      <webgen:ParameterDescription>
         <webgen:name>geom</webgen:name>
         <webgen:type>FeatureCollection</webgen:type>
         <webgen:attribute>
            <webgen:name>GEOMETRY</webgen:name>
            <webgen:type>GEOMETRY</webgen:type>
            <webgen:supportedvalues>
               Point,LineString,Polygon
            </webgen:supportedvalues>
         </webgen:attribute>
         <webgen:description>
            layer with geometries to buffer
         </webgen:description>
      </webgen:ParameterDescription>
      <webgen:ParameterDescription>
         ... buffer width, type double...
      </webgen:ParameterDescription>
   </webgen:InputParameters>
   <webgen:OutputParameters>
      <webgen:ParameterDescription>
         ... result, type FeatureCollection ...
      </webgen:ParameterDescription>
   </webgen:OutputParameters>
</webgen:Interface>
```

For input and output parameters such as collections of map features (FeatureCollection) the data schema can be specified with the `attribute` tags. This data schema contains specifications about the geometry and attributes of the needed input data. The `supportedvalues` tag specifies which geometry types can be used with this service.

With this data schema information clients can prepare the data to conform to the needs of the service before it is sent. An example would be a service's requirement of an attribute named "class" containing the object class of the sent map features. Thus the service descriptions are containing syntactical specifications about the input and output data types but also some semantical specifications designating the interoperability of the used data schemata.

When executing a service the communication between client and server components is realized using standard HTTP (Hypertext Transfer Protocol) calls for transferring the parameters and results as XML data. This assures interoperability as HTTP communication and XML parsers are available for every major platform. So client and/or server components can also be implemented on other platforms. For both calls to a service as well as responses from the server exactly the same data format and structure is used, thus only one set of XML encoders and parsers are needed for every platform. The following XML code example in listing A.2 shows a typical call to a buffering service.

Listing A.2: WebGen protocol example

```
<?xml version="1.0" encoding="UTF-8" ?>
<webgen:Request xmlns:webgen="http://www.webgen.org/webgen"
xmlns:gml="http://www.opengis.net/gml" algorithm="SimpleBuffer">
   <webgen:Parameter name="geom" type="FeatureCollection">
      <FeatureSchema>
         ... attributenames and types ...
      </FeatureSchema>
      <FeatureSet>
         ... arbitrary number of features
             with geometries and attributes ...
      </FeatureSet>
   </webgen:Parameter>
  <webgen:Parameter name="width" type="DOUBLE">
   25.0
  </webgen:Parameter>
</webgen:Request>
```

As can be seen in the XML example the tag Request can have an arbitrary number of `Parameter` entries. These parameters can be of various types and are basically the containers used to transfer the data etc. to the server and the response back to the calling client. Data types for the parameters may include simple types such as strings, doubles or integers but also complex types such as geometries, map features with attributes, collections of map features, matrices or triangulation and graph structures encoded in XML. New data types may be added to the system by providing an encoder and a parser for the new type and including it into the client and server components.

Client and Server Components

An overview of the WebGen client components is shown in figure A.1. Clients for the WebGen framework take the parameters, which are supplied by the user and transfer them to the generalization service (see also figure A.2). For selecting a service and retrieving the service's interface they query the registry. The server component offers generalization services, e.g. algorithms that are executed on the server, to clients or other servers (cf. figure A.1).

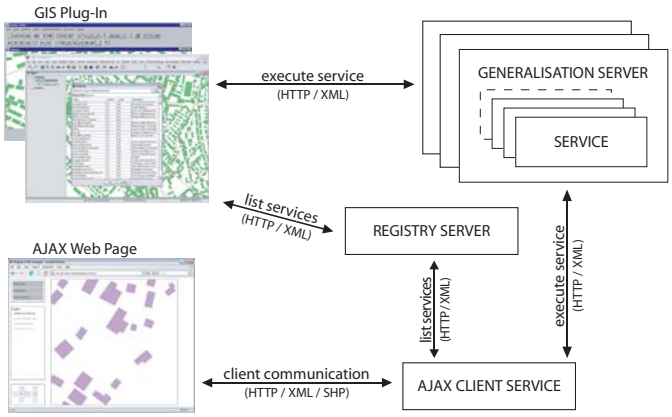


Figure A.1: WebGen client components

Accessing the services is currently possible through a plug-in for the JUMP Unified Mapping Platform (2006), through a browser based AJAX web page (Asynchronous JavaScript and XML) and through a prototype plug-in for the Clarity generalization software by 1Spatial, Ltd. (2007). The client plug-in offers the configuration and selection of the desired service. The data to be processed by the service together with the parameters for the algorithm are encoded and sent to the service. The result, returned by the service, is then decoded and presented in the client. A client with similar functionalities could also be developed for other desktop GIS that have an API for adding functionalities.

The first implementation of a generalization server for the WebGen framework was written as JAVA software for the Tomcat application server (Apache, 2007). Algorithms written by researchers in JAVA can be made available directly. Algorithms written in other programming languages (e.g. C, C++ or Visual Basic) or being available as executable programs can also be included with full functionality. To that end these algorithms must be installed as executable on the server and they must be able to read the data and parameters from a standard format such as Shapefiles. Internally the JTS Topology Suite (2006) is used for the geometry representation. JTS conforms to the Simple Features Specification for SQL, published by the Open GIS Consortium (1999).

The server implementation foresees a single endpoint on every generalization server (cf. figure A.2). This endpoint is a JAVA Servlet hosted by the Tomcat server. As every execution call to a server contains the name of the service the service dispatcher can decode the parameters and pass them to the called service. Every service must be represented by a separate JAVA class file on the server respecting a common interface, which can be called by the service dispatcher. The service's functionalities, e.g. the algorithm, can be executed either directly in this class, in another JAVA program (e.g. from an existing algorithm library) or also in a separate executable.

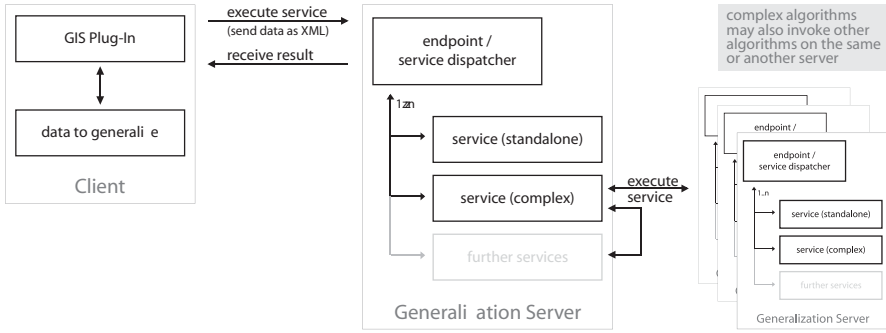


Figure A.2: WebGen server components

Usually the services have a stateless lifecycle. Meaning that they are instantiated upon being called, they use only parameters from the call, send the result back to the caller and then cease to exist. Such a *stateless service* can not remember any previous parameters or other settings. *Stateful services* however can store parameters and settings when being called multiple times. This is done by initializing a session of the service for every service user. The session then stores the parameters, e.g. the map data, which are transferred to the service. During the lifetime of the service the data can be queried and modified. Thus repeated uploading of massive amounts of data to the service is avoided.

As a container for all sorts of parameters of a workflow a stateful transaction server was implemented. Here a session is initialized once and the map data (features) and other parameters which will be treated in the subsequent workflow can be uploaded. Following that the data can be accessed and modified by other services.

Every service may also make calls to other services on the same or on another generalization server. Usually this is done in a stateless manner. Thus e.g. a processing service which needs a buffer around a point, sends the point geometry to the service and receives the buffer's polygon geometry as a result. The stateful use of such a support service makes sense when for example a support service is used by another service several times with basically the same data and only parameters changed. An example is a support service for Delaunay triangulations which can be initialized once with the geometries and then only be queried by requesting e.g. the shortest or longest edge or modified by adding or removing nodes.

A WebGen generalization server is installed at the Geographic Institute of the University of Zürich and available under `http://webgen.geo.uzh.ch/` hosting all the algorithms which were developed during the project DEGEN. The WebGen client plug-in for the JUMP Unified Mapping Platform can also be downloaded from this address.

Appendix B

Comment on OGC Web Processing Service (WPS)

This comment is a reply to the OGC request for public comments about the proposal of OGC Web Processing Services (WPS). Such a standard could also serve as basis for a generalization services framework comparable and compatible to the WebGen framework. The specification for the WPS standard however were much to open and unspecific at this date in order to be usable as a standard e.g. for generalization services.

OGC Request 28: Web Processing Service (WPS): Request for public comments
03. February 2006

PART A

1. Evaluator:
Moritz Neun
Department of Geography, GIS Division, University of Zurich, Switzerland
neun@geo.unizh.ch
www.geo.unizh.ch/ neun/
2. Submission: OGC Request 28: Web Processing Service (WPS): Request for public comments

PART B

Specification Section number: GENERAL
Criticality: EDITORIAL
Comments/justifications for changes: COMMENTS
We have been experimenting on our own now for more than a year with web services for the execution of generalization algorithms. In [1] we show a classification of services and a first prototype for a generalization processing service called WebGen which is based on SOAP. Our proposal includes one or more central registry servers which are responsible for the listing of available services (GetCapabilities) and for the deployment of the service descriptions (DescribeProcess). See for more detail in [2] where this technique is proposed to be used to share algorithms in research, e.g. for enhancing comparability and making the reuse of existing algorithms easier. Although our web services were mainly intended for generalization the concepts are also applicable for Web Processing Services in general.

In general we make a distinction between the two service concepts middleware and standalone service (research platform). While the middleware is merely intended for the access of generalized data from e.g. WFS through such a middleware service, the standalone service gets the data from the service user who would e.g. upload the data directly to the server. The WPS specification seems to be a hybrid between those two concepts. In [3] first thoughts about not only using standard geodata (which can be expressed by e.g. GML) were presented. The intention was to have web services offering e.g. supporting datastructures such as triangulations or hierarchies. This goes beyond the abilities of e.g. the proposed buffering service which in fact has simple features as input and output.

Specification Section number: GENERAL
Criticality: MAJOR
Comments/justifications for changes: COMMENTS
The WPS specification is much too unspecific and open. Thus I don't see it as a real standard, it's merely a proposal how one could develop a web service for processing geographic data. This specification does neither define the data format nor the data schema or semantics. Consequently, accessing a WPS is very difficult, because the client has to be capable of so many different data formats. Thus a generic client program which could be used for accessing WPSes is probably almost unimaginable and the publish-find-bind concept of web services does not apply for WPS. But most of the geo operators have quite similar input and output requirements. Looking at vector data, as we use in our work on generalization algorithms and services, the map objects to process consist usually of a geometry which is a simple feature and a couple of attributes. Thus for most Web Processing Services a more standardized (structurally, and semantically) data format could be used. For most of the standard generalization algorithms we looked at in our WebGen project [2] this was completely sufficient. The more specific data formats [3] can still be left more open.

The WPS DescribeProcess does not include any geospatial specifics. Thus there is no advantage over e.g. WSDL. The WebGen prototype basically uses the WSDL concept and extends it with the ability to define the data schema (geometries and attributes) supported by the processing service. Thus the WebGen client program can take care of sending the data in the right schema. The data is then coded directly onto the SOAP communication as GML fragments. This is very straightforward and creates less overhead.

The WPS GetCapabilities functionality is not sufficient as it lists only the available services on one server. Thus for the retrieval of a specific service the server address must be known. The service discovery has to be done by the user e.g. in a web-browser and can not be included in the process. So, again the publish-find-bind rule of web services is not fulfilled as there are no yellow-pages (e.g. UDDI) where all services are registered. The WebGen prototype includes a registry server (can also be distributed for backup) where all services are registered, see [1] and [2]. This server holds the descriptions of the services and their endpoints where they can be reached. Every service provider just has to enter his/her service into this database. With this concept we try to fulfill the publish-find-bind concept, where mostly generic client programs can retrieve the services from the registry and demand the right data from the user.

Concluding the general comments I am absolutely in favor of a standard for Web Processing Services, but the proposed specification is far too open and is missing much geospatial relevance. With our WebGen prototype we defined a simpler, more generic and straightforward solution but with similar concepts (except the registry server). This could provide some improvements to the current WPS specification no matter which protocols are used.

REFERENCES

[1] Burghardt, D., M. Neun and R. Weibel. 2005. Generalization Services on the Web - A Classification and an Initial Prototype Implementation. Proceedings AutoCarto 2005, Las Vegas, USA.

[2] Neun M. and D. Burghardt. 2005. Web Services for an Open Generalisation Research Platform. Proc. 8th ICA WORKSHOP on Generalisation and Multiple Representation, A Coruña, Spain.

[3] Neun M., D. Burghardt and R. Weibel. 2006. Spatial structures as generalisation support services. Proc. Joint ISPRS Workshop Workshop on Multiple Representation and Interoperability of Spatial Data, February 22-24, Hannover, Germany.

Appendix C

Complete Publication List

The research done in the course of this thesis resulted in a set of additional publications that have not explicitly been introduced in this volume. The listing is in chronological order and the additional publications are marked with a star *.

* **Neun, M., R. Weibel and D. Burghardt (2004).** Data Enrichment for Adaptive Generalisation. In *7th ICA Workshop on Generalisation and Multiple Representation*, Leicester, UK.

* **Burghardt, D., M. Neun and R. Weibel (2005).** Generalization Services on the Web – A Classification and an Initial Prototype Implementation. In *Proceedings AutoCarto 2005*, Las Vegas, USA.

* **Neun M. and S. Steiniger (2005).** Modelling Cartographic Relations for Categorical Maps. In *Proceedings of the XXII International Cartographic Conference*, A Coruña, Spain.

* **Neun M. and D. Burghardt (2005).** Web Services for an Open Generalisation Research Platform. In *Proceedings of the 8th ICA Workshop on Generalisation and Multiple Representation*, A Coruña, Spain.

* **Edwardes, A., D. Burghardt and M. Neun (2005).** Interoperability in Map Generalisation Research. In *Proceedings of International Symposium on Generalization of Information*, ISGI, Berlin, Germany.

Burghardt, D., M. Neun and R. Weibel (2005). Generalization Services on the Web – Classification and an Initial Prototype Implementation. *Cartography and Geographic Information Science*, Volume 32, Number 4, October 2005, pp. 257-268(12).

* **Neun, M., D. Burghardt and R. Weibel (2006).** Spatial structures as generalisation support services. In *Proceeings Joint ISPRS Workshop Workshop on Multiple Representation and Interoperability of Spatial Data*, February 22-24, Hannover, Germany.

Burghardt D. and M. Neun (2006). Automated sequencing of generalisation services based on collaborative filtering. In: M. Raubal, H. J. Miller, A. U. Frank and M. Goodchild (eds): *Geographic Information Science. 4th Inernational. Conference on Geographical Information Science (GIScience 2006)*, IfGIprints 28, pp. 41-46.

* **Neun, M. (2006).** Web Services für die kartographische Generalisierung. *Newsletter e-geo.ch.* http://www.e-geo.ch/docu/newsletter/Newsletter_2006_15.pdf (accessed 04/2007)

Edwardes, A., D. Burghardt and M. Neun (2007). Experiments to build an open generalisation system. In W. Mackaness, A. Ruas, & T. Sarjakoski (Eds.), *Generalisation of geographic Information: Cartographic Modelling and Applications* chapter 8, (pp. 161-175). Amsterdam: Elsevier Science.

*** Bobzien, M., D. Burghardt, I. Petzold, M. Neun and R. Weibel (in press).** Multi-Representation Databases with Explicitly Modelled Intra-Resolution, Inter-Resolution and Update Relations. *Cartography and Geographic Information Science*.

Neun M., D. Burghardt and R. Weibel (in press). Web Service Approaches for Providing Enriched Data Structures to Generalisation Operators. Accepted for *International Journal of Geographical Information Science*.

Neun M., D. Burghardt and R. Weibel (submitted). Automated Processing for Map Generalization with Modular Operator Services. Submitted to *GeoInformatica*.

Appendix D

Curriculum Vitae

MORITZ DOMINIKUS ANDREAS **NEUN**

born December 21st, 1978, in Augsburg, Germany

Education

1998	Graduation from high school (Vöhlin Gymnasium Memmingen, Germany). Primary subjects: physics, history, French, mathematics.
1999 - 2002	Diploma studies of Computer Science, with minor subject Geography, at the University of Fribourg (Switzerland), bilingual in German and French. Specialization in internet technologies and telecommunication. Term papers about SOAP, wireless communication and customer relationship management.
2002 - 2004	Transfer from diploma program to MSc in Computer Science program at the University of Fribourg, minor subject Geography remains unchanged.
2004	Graduation with Master of Science (MSc) in Computer Science from the University of Fribourg (Switzerland). Master Thesis: *"WebDiP - Web based Decision Process tracing and data mining"*
2004 - 2007	PhD student at the Geographic Information Systems Division, Department of Geography, University of Zurich (Switzerland). Title of the thesis: *"Data Enrichment for Adaptive Map Generalization Using Web Services"*, advised by Dr. Dirk Burghardt and Prof. Dr. Robert Weibel.