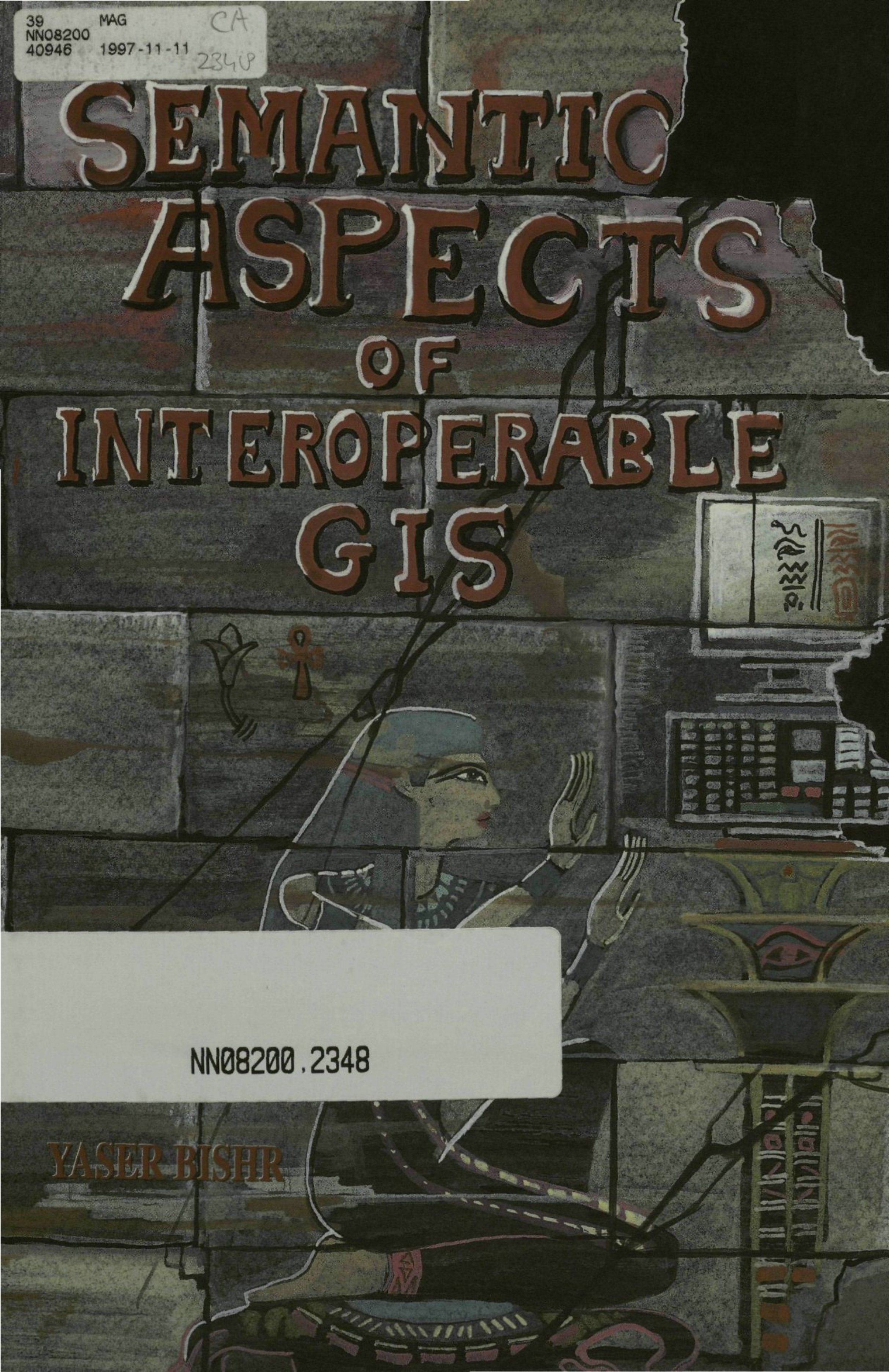
SEMANTIC ASPECTS OF INTEROPERABLE GIS



YASER BISHR

Semantic Aspects of Interoperable GIS

Yaser Bishr

Thesis
to fulfill the requirements for the degree of doctor
on the authority of the rector magnificus,
Dr. C.M. Karssen,
to be publicly defended
on Monday, 24 November 1997,
at 16:00 hours in the Auditorium
of Wageningen Agricultural University

Abstract

Yaser Bishr, 1997. Semantic Aspects of Interoperable GIS. Ph.D. Dissertation.

An increasing number of geospatial applications require information which is scattered in several independent geographic information systems. One of the main objectives of geographic information infrastructure, GII, is to provide a political, institutional, economic, and technical platform to share information. The focus of this thesis is on the technical aspects of the GII. The thesis aims at providing a mechanism to share information seamlessly among distributed, heterogeneous, geospatial information systems.

Sharing information may improve decision making and reduce the cost of data collection. In general, retrieving information from distributed databases involves two steps. In the first step users search for relevant information resources in a network of information providers. In the second step users request data from the information resource.

With regard to the first step, a model to search for the relevant information resources is presented. The model is called the resource discovery model, RDM. It provides a reference model to structure the metadata of the information resources in a tree of interrelated resources.

In the second step, interoperability allows communication among heterogeneous, distributed, information systems. Interoperability is the ability of two or more systems to exchange geospatial information and to make mutual use of the information that has been exchanged.

The research identifies two perspectives to interoperability; these are the data modeling perspective and the system architecture perspective. In relation to data modeling, three types of heterogeneity arise: syntactic, schematic, and semantic. The semantic heterogeneity occurs due to differences in the definition of classes, the definition of class intension and the geometric description. This set of definitions is called context information. The semantic heterogeneity is the main factor for the schematic and the syntactic heterogeneity.

The schematic heterogeneity pertains to the differences in the class hierarchies and the attribute structure of database schemas. The syntactic heterogeneity occurs due to the differences in the constructs used to model relationships among classes and attributes, object geometry, and topologic relationships.

To provide interoperability among different GIS applications, it is necessary to resolve the semantic, schematic, and syntactic heterogeneity.

In this thesis, a model for information sharing is presented. The model is called the semantic formal data structure, SFDS. It consists of three layers, in which each layer is intended to resolve a specific type of heterogeneity. The model provides a method for loading semantics, that is the context information, into database schemas. The first layer of SFDS is the syntactic layer, in which the formal data structure, FDS, is adopted. The second layer of SFDS is the schematic layer, in which the concept of federated databases is adopted. A reference model for the federated schemas is presented. The implementation of SFDS and RDM is related to the system architecture perspective for interoperability, and is discussed in this thesis.

A comparison of the implementation of RDM, which is a clearinghouse, with other implementations has proven that a consistent abstract model is required to maintain and ensure the consistency of the contents of the clearinghouse as well as to improve the results of the search for information resources.

The three-layers approach adopted in SFDS has proven adequate to resolve the three types of heterogeneity. The implementation of SFDS, known as the semantic translator, has shown that it should be dedicated to a single application domain, to simplify its practical implementation and maintenance. In this case databases can have several semantic translators installed, each being specific to an application domain. For example, one database may have a semantic translator to exchange road network information, another to exchange soil information, etc.

Keywords: context information, federated databases, interoperability, information resources, multi-level decision support systems, ontology, proxy context, context, resource discovery, schematic heterogeneity, semantic heterogeneity, semantic similarity, semantic translators, syntactic heterogeneity.

SAMENVATTING

Bishr, Y. A., 1997. Semantische aspecten van interoperabele ruimtelijke informatiesystemen. Proefschrift ter verkrijging van de graad van Doctor.

Een toenemend aantal ruimtelijke toepassingen vereist informatie die in verschillende geografisch onafhankelijke informatiesystemen verspreid aanwezig is.. Eén van de voornaamste doelstellingen van de geografische informatie infrastructuur, GII, is te voorzien in een politiek, institutioneel, economisch en technisch platform om informatie te delen. Dit proefschrift richt zich vooral op de technische aspecten van de GII. Het stelt zich ten doel een techniek te ontwikkelen waarmee informatie consequent tussen her en der gedistribueerde, ongelijksoortige, ruimtelijke informatiesystemen kan worden uitgewisseld.

Het delen van informatie kan besluitvormingsprocessen verbeteren en de kosten van het verzamelen van gegevens beperken. In het algemeen zijn er voor het ontsluiten van informatie uit gedistribueerde databases twee stappen nodig. Bij de eerste stap zoeken gebruikers naar relevante informatiebronnen in een netwerk van informatieleveranciers. Tijdens de tweede stap vragen gebruikers gegevens op uit de informatiebron.

Met betrekking tot de eerste stap wordt een model geïntroduceerd om naar de relevante informatiebronnen te zoeken. Het model wordt het *resource discovery model*, RDM, genoemd. Het biedt een referentiemodel dat de metadata van de informatiebronnen in een boomstructuur ordent. In de tweede stap maakt interoperabiliteit communicatie mogelijk tussen heterogene, gedistribueerde informatiesystemen. Interoperabiliteit is het vermogen van twee of meer systemen om georuimtelijke informatie uit te wisselen en onderling gebruik te maken van de informatie die op deze wijze is uitgewisseld.

Het onderzoek beziet interoperabiliteit vanuit twee gezichtspunten; vanuit het perspectief van de gegevensmodellering (*data modeling perspective*) en van de systeemarchitectuur. Met betrekking tot de gegevensmodellering doen zich drie typen heterogeniteit voor: syntactische, schematische en semantische. De semantische heterogeniteit treedt op door verschillen in de definitie van klassen, de definitie van klasse intensie en de geometrische beschrijving. Deze verzameling definities wordt contextuele informatie genoemd. De semantische heterogeniteit is de voornaamste factor voor de schematische en de syntactische heterogeniteit.

De schematische heterogeniteit heeft betrekking op de verschillen in de klasse hiërarchieën en de attribuutstructuur van databaseschema's. De syntactische heterogeniteit doet zich voor door de verschillen in de concepten die gehanteerd worden om betrekkingen te modelleren tussen klassen en attributen, geometrie van voorwerpen, en topologische betrekkingen.

Om interoperabiliteit tussen verschillende GIS toepassingen te realiseren is het noodzakelijk een oplossing te vinden voor de semantische, schematische en syntactische heterogeniteit.

In dit proefschrift wordt een model voor *information sharing* geïntroduceerd. Het model wordt de semantische formele gegevensstructuur, *semantic formal data structure*, SFDS genoemd. Het bestaat uit drie lagen, waarin elke laag is bedoeld om een oplossing te vinden voor een specifiek type heterogeniteit. Het model voorziet in een methode om semantiek, d.w.z. de contextuele informatie, in database schema's te laden. De eerste laag van SFDS is de syntactische laag, waarin de formele gegevensstructuur, FDS, wordt toegepast. De tweede laag van SFDS is de schematische laag, waarin het concept van overkoepelende (*federated*) databases wordt toegepast. Een referentiemodel voor de overkoepelende schema's wordt geïntroduceerd. De implementatie van SFDS en RDM houdt verband met de systeemarchitectuur en wordt in deze dissertatie besproken.

Uit een vergelijking van de implementatie van RDM, welke een *clearinghouse* is, met andere implementaties is gebleken dat een samenhangend abstract model is vereist, niet alleen om de samenhang van de inhoud van het *clearinghouse* in stand te houden en veilig te stellen, maar ook om de resultaten van het zoeken naar informatiebronnen te verbeteren.

De drie-lagen benadering zoals deze is toegepast bij SFDS is geschikt gebleken om een oplossing te bieden voor de drie typen heterogeniteit. Uit de implementatie van SFDS, bekend als de semantische vertaler, is gebleken dat deze volledig toegewezen dient te worden aan één enkel toepassingsgebied, teneinde de praktische implementatie en het onderhoud ervan te vereenvoudigen. In dit geval kunnen databases verschillende semantische vertalers installeren, waarbij elke afzonderlijke vertaler specifiek is voor een toepassingsgebied. Zo kan bijvoorbeeld de ene database over een semantische vertaler beschikken om informatie over een wegennet uit te wisselen, de andere om informatie over de bodemgesteldheid uit te wisselen, etc.

Trefwoorden: contextuele informatie, *federated* databases. interoperabiliteit, informatiebronnen, *multi-level decision support systems*, ontologie, *proxy context*, bron detectie, schematische heterogeniteit, semantische heterogeniteit, semantische similariteit, semantische vertalers, syntactische heterogeniteit

Acknowledgments

The completion of a Ph.D. thesis is a once in a life time experience. I was always aware of the hard time and the stress that it can inflict on anybody who attempts to pursue it. Lately I learned that no matter how organized one might be, drawing up a Ph.D. is accompanied by a time of disorder, stress, and sleepless nights. I also realized that people around me are not left intact.

I would like to express my genuine gratitude to my supervisors; this work was not possible without them. Their scientific contribution was unlimited. They have provided me with a flux of knowledge about Geoinformation sciences. They have provided me with confidence and strength to push forward and refine my ideas in a new and difficult field such as interoperability of GIS. Professor Dr. Martien Molenaar, my supervisor, did not spare any effort to guide and assist me throughout my research, especially the data modeling and the foundation of GIS theory. He always believed in my research and gave me strength to continue during the most difficult times. I am indebted to my co-supervisor, second father, and my friend Dr. Moustafa Radwan. He honored me with his guidance and knowledge during the past six years, especially in the system architecture part of my work. I am grateful to have prominent experts in the field of GIS as my supervisors, who showed patience and gave constructive criticism to get my thesis in its final shape.

I would also like to acknowledge my debt to Dr. Radwan's family who provided me with a haven during the most stressful times since I came to Holland in 1992. Mrs. Radwan made me another member of the family. I would like to especially thank the whole family for their moral support and I value their friendship which I will always honor.

I appreciate the advice and assistance of my colleagues both in ITC and elsewhere. I am very appreciative to Prof. Dr. Werner Kuhn, Institute for Geoinformatics (IfGI), University of Muenster, Germany, who showed a lot of interest in my work and always managed to find time in his busy schedule to actively share with me his experience in the semantic aspects of data sharing. Prof. Ir. R. Groot occasionally give me some insights on management aspects of goeinformation infrastructure. Dr. Rolf de By provided valuable and critical comments; he spent a lot of effort to assist me in sharpening the computer science aspects of my research. Dr. Dianne Richardson from CCRS, Canada, shared her valuable ideas and experience in data modeling. With my colleagues Dr. M. Pilouk, Dr. O. Kufoniyi, and Dr. W. Peng I had interesting discussions on 3D GIS, data modeling, and automated generalization.

My debt is also due to ITC, the place where I have spent 6 years. I thank ITC for giving me the chance to pursue my Ph.D. research from the internal research fund. I specifically thank Professor Dr. A Meijerink and Dr. E. Kosters, who did not spare any effort to provide the research funds. Mr. A. Bakker, the administrative director of the department of Geoinformatics, made sure that the finance for my trips to conferences, scientific meetings, and research requirements were always available.

During my research, I was fortunate to be involved in supervising five intelligent M.Sc. students who graduated with distinction. Each implemented different aspects of my research; their contribution is very much appreciated and they provided me with feedback to further refine the concepts. Mr. E. Espinoza implemented parts of the multi-level decision support system, discussed in chapter 2. Mr. T. Mabote implemented a prototype concept from chapter 4. Mr. Matin implemented a component for semantic data sharing presented in chapters 5 and 6. Mrs. N. Feriha partly implemented the concept of hierarchical clearinghouse, described in chapter 7. Mr. Jatin is an experienced programmer who greatly assisted me in developing the prototype of Chapter 8.

The following people were involved in the final stage of the thesis. My talented childhood friend A. Khalil spared hours from his work to design the cover page. My good friend Annet Pril spent long evenings after her work to take care of the layout and the references of the thesis. I can not remember the number of times she had to proofread it, I wonder if she deserves a Ph.D. herself!! Drs. Wan Bakx helped in producing the images of the user interface of SemWeb. Drs. J. Kuipers kindly edited this thesis. Mrs. Anneke Homan relieved me from the administrative process and finalized the production of the thesis. Saskia Tempelman the secretary of the department of Geoinformatics took care of my paperwork in the department for three and a half years. Mr. H. Scharrenborg kindly prepared the color production of the cover page.

My special thanks go to my professors in the Shoubra Faculty of Engineering, Cairo. Also, I thank my friends, the best one can have. They are a great group who have been continuously calling me, and made my holidays in Egypt relaxing and cheerful. I was also fortunate to have great friends and Ph.D. colleagues in Holland.

I also owe a great deal of debt to my special friend Dr. Christine Pohl who was always there for me and always found words to encourage me. During the last few days before submitting my thesis she especially flew from Madrid to assist me. Thank you!!

Last but not least, I could have completed this thesis without my great loving family. I cannot find the proper words to express my sincere appreciation to my parents. They have accepted my absence for six long years and helped me in seeking my ultimate scientific goal. My father supported me mentally and financially and allowed me to live here comfortably. My loving mother visited me several times and embraced me with her care and love. She reminded me how much I missed her nice warm meals. My sisters Amal, Riham, Nihal, my brother Mohammed and my brother in law, Dr. Sherif, visited me regularly, and never stopped calling me no matter where they were in the world. I love you all!!

To those whose contributions I have neglected to note here from sheer failure of memory or character, please accept my apologies. I owe you all a great debt.

11 October, 1997

Yaser Bishr

Table of Contents

List of Figures

List of Tables

CHAPTER

1 Framework and Objectives

"If you steal from one author, it's plagiarism; if you steal from many, it's research."

Wilson Mizner (1876–1933), *Quoted in: Alva Johnston, The Legendary Mizners, ch. 4 (1953).*

1.1 Introduction

Through the 1970s and the early 1980s most GIS applications were considered islands of information. They were self-contained independent systems, where geospatial data were digitally captured, stored, analyzed, and displayed. Data were rarely acquired from other digital sources due to the proprietary nature of the file formats. With the advances in information technology and the growing demands from GIS users to obviate the bottleneck and the high cost of data capture, users began to exchange and transfer information from one system to the other. Such transfer was accomplished either by special purpose translators or by means of a neutral format which could be understood by the source and the target systems. The rapid development of data capture techniques, e.g., scanners, satellites, automatic digitizing, etc., led to an increase in the availability of digital data. The problem, then, has become not how to capture data, but to find out where the most reliable data exist, and how to retrieve them in an acceptable form.

The development of computer networks during the late 80s and the 90s provided users with the possibility of linking spatially distributed computers. They realized the effectiveness of sharing information across computers, via networks. Whether the information was transferred through networks or through any other media, the transfer was characterized by the fact that it was batch-oriented, so that an entire information set was converted and transferred on the file level.

1.2 Problems of Information Sharing

Consider two groups of earth scientists. One is concerned with soil conservation of a particular river basin, and the second manages land-use planning projects at the same basin. The soil conservation group collects detailed soil information, which is also needed by the land-use management group. Why shouldn't the soil conservation group and the land-use planning group share their information instead of collecting the same data again [Buehler et al., 1996]? There are several reasons:

1) The two groups might use two different GIS software platforms that produce two different digital formats and have different representation and analytical capabilities. This is known as the "information transfer" problem. Today the translation could be made by using file converters supplied by the software vendors. However, information is likely to be lost during this process. For example, loss of information would occur during the conversion of information stored in Arc/Info to Intergraph, because there is no perfect union between the functionality of the two systems.

2) Even if the two groups run the same GIS platform, and hence have the same database paradigm, e.g. relational, they might have different conceptual data models, different data collection schemes, and different quality parameters. In this case information transfer from one group to the other requires mapping between the corresponding data models.

3) Institutional, economic, and legal obstacles might limit the freedom of information sharing.

1.3 Information Sharing in a Geo-information Infrastructure

The provision of systems and mechanisms for the transfer of information is closely related to a field in geoinformatics known as geographic information infrastructure, GII. GII can be described as a set of institutional, technical, and economic arrangements to support the availability of relevant, up-to-date, and integrated geoinformation, timely and at an affordable cost. Figure 1-1 shows such a utility at an abstract level [Radwan et al., 1996]. On the right-hand side of the figure are the independent databases dedicated to local applications. These databases can be linked in a client-server architecture. On the left-hand side of the figure, the federated database approach provides global data models to facilitate seamless interaction. The question marks in the figure show the problems related to information sharing in a GII. The problems can be subsumed under two main categories:

1) *Political, Institutional and Economic problems*: The provision of information for the public requires legal considerations (e.g., copyright), proper institutional organization and data access rules, as well as pricing schemes.

2) *Technical problems*: These include devising techniques for the provision of up-to-date inventory of the available data, mechanisms for seamlessly sharing information, update and consistency constraints.

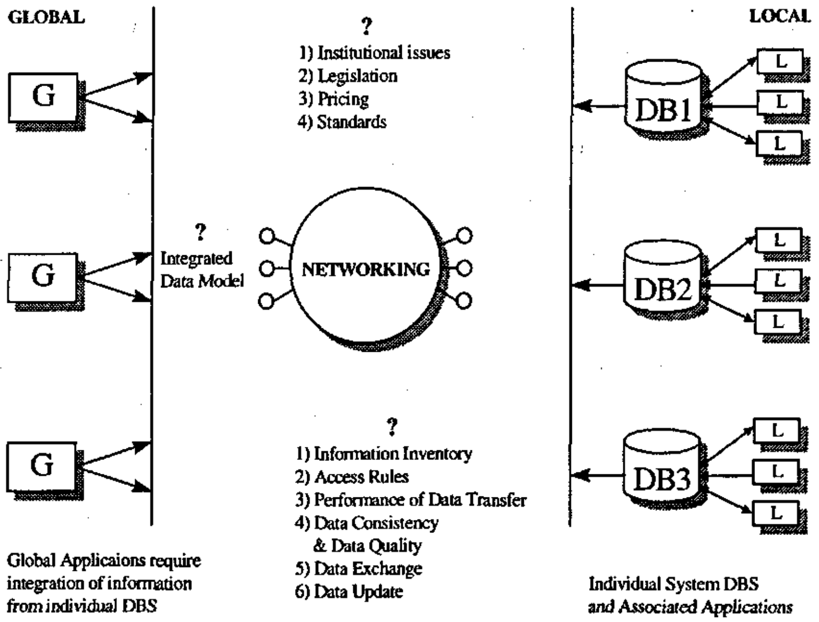

GLOBAL
?
1) Institutional issues
2) Legislation
3) Pricing
4) Standards
LOCAL
G
G
G
DB1
DB2
DB3
L
?
Integrated Data Model
NETWORKING
?
1) Information Inventory
2) Access Rules
3) Performance of Data Transfer
4) Data Consistency & Data Quality
5) Data Exchange
6) Data Update
Global Applicaions require integration of information from individual DBS
Individual System DBS and Associated Applications

Figure 1-1 A conception model for a geo-information infrastructure.

With reference to the technical issue, the ongoing research and development mainly focus on providing interoperability among platforms (hardware and database management systems). Several solutions are available in the market which provide interoperability among database management systems, DBMS, e.g., ODBC, ORACLE, and JET [Geiger, 1995; ORACLE, 1992; Fawcette, 1996]. However, little has been done on the application side, particularly in the field of GIS. This is due to the fact that it is difficult to resolve the differences between their conceptual and logical models.

From the above discussion it can be concluded that the issue of information sharing may be viewed from two different perspectives:

- *System perspective*: where the focus is the interaction among the different components of the underlying system.
- *Data modeling perspective*: where the focus is to resolve the differences in the conceptual and the logical models of the databases of the underlying system.

1.4 Framèwork of the Study

This research is conducted under the umbrella of a broader internal ITC research project which aims at developing a multilevel decision support system, MLDSS, for environmental decision-making. Environmental decision-making usually requires spatial and non-spatial information from different disciplines. Geographic information systems are capable of handling both types of information in an integrated environment. Incorporating GIS within a decision support system can increase its functional capabilities and will allow decision makers to analyze the impact of their decisions.

A decision support system (DSS) refers to a collection of computerized technologies whose objective is to support managerial work, particularly decision making [Turban, 1993]. DSS provides a more thorough understanding of the problem domain, and as a result leads to more environmentally and economically sound decisions [Wilde, 1994; Buhyoff et al., 1994].

In most cases, the information needed to pursue environmental analysis is required as fast as possible in order to cope with the rapidly changing environment [Radwan et al., 1996]. High costs of information acquisition to support such complex applications emphasizes the need for sharing information. One of the main objectives of the aforementioned project is to show how information sharing can improve the process of environmental decision making.

1.5 Scope of the Research

This research is mainly concerned with the technical aspects for the development of geo-information infrastructure. More precisely, the focus is on the data modeling perspective for information sharing. This can be achieved by developing a mechanism for seamless information sharing. The system architecture perspective is of secondary importance. A system was implemented to evaluate in how far the first objective has been attained. The term seamless, as used here, means that the provision of a system is required that makes it unnecessary for users to understand and explore the data model of the other databases. Moreover, users do not have to commit themselves to a particular data model, while sending queries or receiving data. This also means that they will not have to understand the contents and semantics of the remote databases.

1.5.1 The Data Modeling Perspective

Geo-information theory should provide us with a syntax to express spatial knowledge and it should include an explanation of how semantics can be loaded on such a syntax. Ter Bekke, defines semantics as the discipline which deals with relationships between words and the things to which these words refer [Ter Bekke, 1992]. In the database modeling domain, semantics is concerned with the study of the meaning and relationship between real world features and database objects.

Geo-information theory defines the conceptual and the spatial models for a wide variety of applications. The common characteristics of the conceptual models are studied to understand how they can best be mapped into selected logical models. Molenaar and others define the foundation of spatial objects in what is called the formal data structure, FDS [Molenaar, 1993; Molenaar, 1994; Molenaar et al., 1994 (a) & (b)]. This foundation forms the GIS syntax. At the lowest level of the syntactic definition, as shown in Figure 1-2, we find the classic data structures, i.e., field and object based approaches. The GIS theory formalizes the topologic relationships amongst fields and objects, uncertainty aspects, and the handling of geometry and topology of fuzzy objects. The theory introduces a consistent framework for object hierarchies, such as generalization and aggregation, which form the building block for schema definitions.

Figure 1-2 shows the semantics to be built onto the syntactic and schematic definitions. Class hierarchies, which are considered a schematic problem in this research, could be viewed as a semantic problem by other researchers [Fang et al., 1991; Fankhauser et al., 1991; Geller et al., 1991]. Class intension, as well as the relationship between instances of the classes and the real world features are considered a semantic problem.

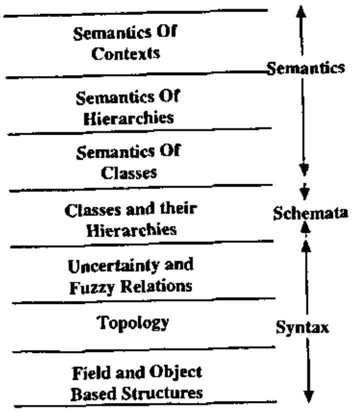


Figure 1-2 Syntactic and semantic definition.

Goh, Madnick and Siegel define context with respect to the view of an application, or in other words, the application semantic view [Goh et al., 1994]. In this study, context is used to refer to both database schema and its semantics. The relationship between contexts is the third semantic level. Defining the semantics of contexts establishes a relationship between different GISs.

1.5.2 System Architecture Perspective

When information stored in spatially distributed information systems is shared, several problems may arise.

- Each database management system has its own functionality and interfaces.
- The databases may be installed on different platforms, which support different network protocols.
- The application protocol, which defines the way two or more databases communicate, may present a problem.

1.6 Research Objectives

The main objective of this research is to develop a theoretical concept for capturing semantics into the database, and to develop a mechanism that accesses these semantics in order to identify spatial objects in a heterogeneous, distributed environment. The concept serves as an extension of FDS, and therefore was built on its foundation [Molenaar, 1994;

Molenaar et al., 1994 (b); Molenaar, 1995]. The requirement for a pragmatic approach to the testing of the developed concept led to the establishment of the secondary objective. The concept is exemplified by a prototype which demonstrated how object identification and transfer can be conducted, based on their semantics. The objectives can, therefore, be outlined as follows:

1) To develop a model, as an extension to FDS, to load semantics on the database objects.

2) To develop a mechanism which accesses these semantics for identification and transfer of spatial objects in a heterogeneous distributed, database environment.

3) To implement the concept, as developed, in a prototype for information transfer between heterogeneous geographic databases, to support MLDSS for environmental decision making.

1.7 Research Approach and Thesis Structure

The research has been conducted in three phases. The first phase sets the framework of the study and identifies the problems of information sharing and interoperability. The first and second objectives, which are related to the data modeling perspective, are addressed in the second phase. The third phase handles the system architecture perspective, and is related to the third objective. In this phase the design and implementation of the prototype is described.

There are two considerations which are important in this research:

- Although the research framework is developing a multi-level decision support system for watershed management, the research focus is on the information sharing problem and not on the environmental decision-making problem.
- The research does not intend to provide a complete formal system for semantic information sharing. Instead, some formalism was introduced to assist in the implementation phase.

1.7.1 Development of MLDSS and Aspects of Information Sharing

In chapter 2, the components of an environmental decision support system with focus on the data modeling issues are studied. Field work is conducted and an experimental prototype is developed. The result of which is a list of guidelines for information sharing which may lead to an understanding of the technical issues which need to be further enhanced.

The objective of chapter 3 is to put the research in the context of the current technological development and to show how this work contributes to science as well as technology. Another objective is to present the overall components of the prototype which implements the concepts developed in this research.

1.7.2 Development of Concepts for Information Sharing

Chapter 4 provides a detailed definition of the abstraction process, as viewed in this research. Heterogeneity among GIS systems is classified. Based on this, an informal explanation of the proposed concept is presented. Chapters 5 and 6 present the concept proposed for semantic translation, known as semantic formal data structure, SFDS. The implementation of SFDS is a semantic translator. Semantic translators provide a medium to interoperate heterogeneous applications. Chapter 7 shows the model which allows users to search and locate information providers (or information resources) as a precursor to information sharing.

1.7.3 Development of the Semantic Translator Prototype

In chapter 3 the prototype is first designed and presented independent of any implementation. Chapter 8 is a detailed description of the implementation of the prototype. Tools which are needed to develop such a system are also presented. Conclusions and issues for future research are introduced in chapter 9.

1.8 Expected Scientific Contribution

- The research presents novel perspectives to achieve interoperability among GIS applications.

- The research extends FDS, which is considered as the syntactic layer, with a schematic and a semantic layer.
- This research can be considered as a first step to provide interoperability among GIS applications. Furthermore it opens the door to further explorations in this area of research.
- It attempts to provide a systematic methodology to develop semantic translators.
- It brings current research of semantic interoperability, which is an active area of research in artificial intelligence, into the field of GIS.

References

Boar, H.B., 1993. *"Implementing Client/Server Computing, A Strategic Perspective"*. McGraw-Hill, Inc. 1993. ISBN 0070062153.

Bordie M.L., 1992 *"The Promise of Distributed Computing and The Challenge of Legacy Information Systems"*. In Proceedings of the IFIP WG2.6 Database Semantics Conference on Interoperable Database Systems (DS-5), Lorne, Victoria, Australia, 16-20 November, (Hsiao, D.K., Neuhold, E.J., and Sacks-Davis, R., eds.), pp. 1-31.

Buehler, K., and McKee, L. (eds.), 1996. *"The OpenGIS™ Guide, Introduction to Interoperable Geoprocessing"*. Open GIS Consortium, Inc., 35 Main Street, Suite 5, Wayland, MA 01778, USA.

Buhyoff, J.G., Miller, P.A., Roach, J.W., Zhou, D., and Fuller, L.G., 1994. *"An AI Methodology for Landscape Visual Assessments"*. In International Journal of Artificial Intelligence for Natural Resources, Vol. 8, No. 1, 1994.

Fang, D., Hammer, J., and McLeod, D., 1991. *"The Identification and Resolution of Semantic Heterogeneity in Multidatabase Systems"*. Proceedings First International Workshop on Interoperability in Multidatabase Systems, 1991, USA, pp. 52-59.

Fankhauser, P., Kracker, M., and Neuhold, E., 1991. *"Semantic vs. Structural Resemblance of Classes"*. SIGMOD Record, Vol. 20, No. 4, December 1991, pp. 59-63.

Fawcette, J.E., 1996. *"The Official Visual Basic Programmer's Journal Guide to Visual Basic 4"*. Que Corporation, 201 W. 103rd Street, Indianapolis, Indiana 46290, USA. ISBN 0 7897 0465 X.

Geiger, K., 1995. *"Inside ODBC"*. Microsoft Programming Series. Microsoft Press, Redmond, Washington 98052-6399. 482 pages. ISBN 1 55615 815 7.

Geller, J., Perl, Y., and Neuhold, E.J., 1991. *"Structure and Semantics in OODB Class Specifications"*. SIGMOD Record, Vol. 20, No. 4, December 1991, pp. 40-43

Goh, C.H., Madnick, S., and Siegel, M., 1994. *"Context Interchange: Overcoming the Challenges of Large-Scale Interoperable Database Systems in a Dynamic Environment"*. In Proceedings of the Third International Conference on Information and Knowledge Management. Gaithersburg, MD, November 1994, pp. 337-346.

Molenaar, M., 1993. *"Object Hierarchies and Uncertainty in GIS or Why is Standardization so Difficult"*. Geoinformation Systems, Vol. 6, No.3, pp. 22-28.

Molenaar, M., 1994. *"A Syntax for the Representation of Fuzzy Spatial Objects"*. (Molenaar, M. and Hoop, S., de, eds.), AGDM'94 Spatial data Modeling and Query Languages for 2D and 3D Applications, Publications on Geodesy - New Series, No. 40, Netherlands Geodetic Commission, Delft, 1994, pp. 155-169.

Molenaar, M., 1995. *"An Introduction into the Theory of Topologic and Hierarchical Object Modeling in Geo-Information Systems"*. Lecture Notes, Department of Land-Surveying & Remote Sensing, Wageningen Agricultural University, The Netherlands, 186 pages.

Molenaar, M., and Janssen, L.L.F., 1994 (a). *"Terrain Objects, Their Dynamics and Their Monitoring by the Integration of GIS and Remote Sensing"*. (Ebner, H. et al., eds.). Spatial Information from Digital Photogrammetry and Computer Vision, proceedings ISPRS Comm.III, Int. Archives of Photogrammetry and Remote Sensing. Vol. 30 part 3, München, Germany, pp. 585-591.

Molenaar, M., and Richardson, D.E., 1994 (b). *"Object Hierarchies for Linking Aggregation Levels in GIS"*. (Welch, R., and Remillard, M., eds.). Mapping and GIS, proceedings of ISPRS Comm.IV, Int. Archives of Photogrammetry and Remote Sensing vol. 30 part 4, Athens Georgia, pp. 610-617.

ORACLE (Cooperative Server Technology for Transparent Data Sharing), 1992. *"ORACLE7 Server Application Developer's Guide"*. Part number 6695-70-1292, December 1992. Oracle Cooperation, 500 Oracle Parkway, Redwood City, CA 94065, USA.

Radwan, M., Bishr, Y, and Paresi, C., 1996. *"The Development of a Geographic Information Infrastructure for Multi-level Decision Making for Environmental Monitoring, Specific to Watershed Management"*. Seminario Internacional Del Medio Ambiente y Desarrollo Sostenible 1996, Bucaramanga, Colombia.

Ter Bekke, J.H., 1991. *"Semantic Data Modeling in Relational Environments"*. Ph.D. Thesis, University of Delft, 141 pages. ISBN 90 900 4132 X.

Turban, E., 1993. *"Decision Support and Expert Systems, Management Support Systems"*. Macmillan Publishing Company, 866 Third Avenue, New York, 10022, U.S.A., 833 pages. ISBN 0 02 422691 7.

Wilde, D.W., 1994. *"Australian Expert Systems for Natural Systems"*. In International Journal for Artificial Intelligence for Natural Resources., Vol. 8, No. 3, 1994. Pp. 3-12.

CHAPTER

2 Environmental Decision Making - A Geoinformation Perspective

"I am, I plus my surroundings and if I do not preserve the latter, I do not preserve myself."

José Ortega y Gasset (1883–1955),
Meditations on Quixote (1914).

2.1 Introduction

Environmental studies are defined as the analysis of the structure, function, and change of interacting ecological entities in a heterogeneous land area composed of groups of organisms (including humans) interacting with their non-living environment [Forman et al., 1986]. Environmental decision making is a multi-disciplinary and multi-level application, which requires information from different sources and hence can benefit from the provision of a mechanism for information sharing.

This chapter explains the results of a research project which was executed during the first stages of this work [Radwan et al., 1993]. The project was initiated at ITC to develop a multi-level decision support system, MLDSS, for environmental decision making. The project attempts to answer two main questions:

- What are the data modeling requirements to achieve semantic interoperability?
- What system is needed to support information sharing amongst disciplines and levels of environmental decision making and what are its requirements?

The outcome of this research project is an MLDSS prototype, which will be briefly explained in this chapter. First an introduction of the inherent complexity of environmental decision making is presented in section 2.2. Section 2.3 explains a real case of environmental decision making. A field study was conducted in the South of Spain, in the region of Andalucía. The objective of this study was to identify the current problems and requirements for environmental decision making from a GII perspective (that is data modeling and system perspectives). The section also shows the process of decision making as observed during the case study. Section 2.4 attempts to answer the question related to data modeling perspective. The second question, which pertains to the system architecture perspective, is tackled in section 2.5. The implementation of the prototype is summarized in section 2.6. The problems and issues identified during the case study and the implementation are outlined in section 2.7. The chapter is then concluded in section 2.8.

2.2 Components of Environmental Decision Making

The need for a new perspective for environmental management became apparent when national economic demands for forest, fish, water, and wildlife resources expanded. Ecosystems, which form the environment, were managed as isolated components although they are interrelated.

Subdivision of land into interrelated systems on different scales is needed for multi-level decision making. An example of the systems is shown in Figure 2-1. Due to the linkages between the systems, a modification of one system may affect the operation of the surrounding ones. Furthermore, the response to the management activities is partially determined by the relationships with the surrounding systems, linked in terms of runoff, groundwater movement, and micro-climate influences [Bailey, 1996]. Understanding these relationships is important for the analysis of the cumulative effects, i.e., action on one scale (local) and effects on another (regional).

Risser mentions that we have to be aware of the land-atmosphere interaction, when land use changes occur on a large spatial scale [Risser, 1993]. Figure 2-1 (a) gives an overview of the relationship between climate, land use, hydrology and soil within a particular watershed. The impact of changing the environmental components on the social and economic components is shown in Figure 2-1 (b).

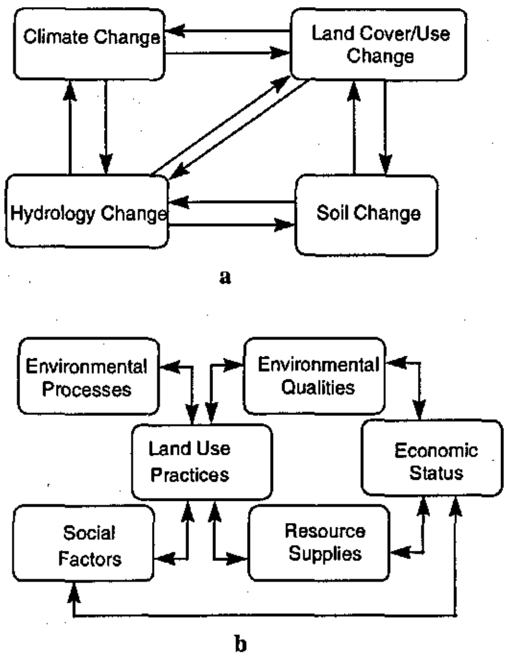


Figure 2-1 Interaction among: a) systems within watersheds; b) components involved in environmental decision making.

In addition to the interrelationship among components, the assessment of the management practices should also consider the level where the sustaining is required (local, regional, or national). Lee and others, have stated that individual management practices by themselves may not cause undue harm. However, taken collectively (at an aggregated level) they may result in degradation and long-term decline in the overall area under consideration [Lee et al, 1993]. The components and levels of environmental decision making can be viewed along the horizontal and vertical axes, respectively. The next section intends to introduce the multi-level issue.

2.3 Levels of Environmental Decision Making

Under the MLDSS research project, a field study has been conducted in the South of Spain, in the region of Andalucía [Bishr et al., 1996 (b); Bishr et al., 1995; Espinoza, 1995]. The intention of the national government is to develop a multi-level decision support system, MLDSS, for watershed management. The aim of the investigation is to address the problems that arise, when the various geo-information systems of differently located organizations, most of which are working at different levels of decision making are linked. At the regional level, each river basin is managed by a hydrologic organization. These organizations report directly to the Ministry of Public Works at the national level. Each hydrologic organization has local offices, which are responsible for the management of the catchments within its underlying basin. Decisions taken by these offices cannot be implemented unless they are agreed upon by their corresponding hydrologic organizations as well as by the Ministry of Public Works. Additionally, decisions taken by the local offices must also be agreed upon by an environmental agency. The environmental agency is an organization independent of the hydrological organization, although it has almost the same organizational structure.

From this organizational structure, the different administrative levels and disciplines are apparent. These are the Ministry of Public Works, the hydrologic organizations and the local offices, and the environmental agency.

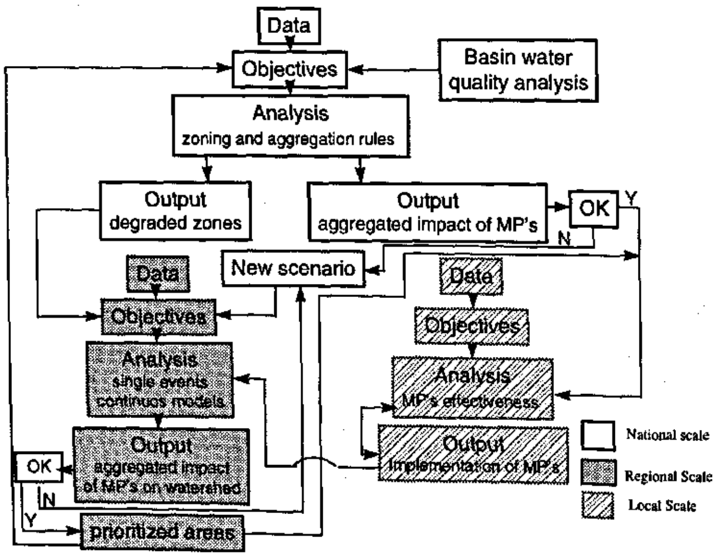


Figure 2-2 Interrelations between the three scales.

Figure 2-2 shows the interaction and relationship among the national, regional, and local levels. They are characterized by an extensive flow of information and decisions among the three levels:

- *At the national level,* decisions concerning the guidelines and constraints for initiating an environmental sustaining project are defined for the whole country. Decisions are taken based on the information provided by the regional level. The information is used to analyze the underlying watershed in order to locate and identify degraded areas. The areas are then ranked according to the rate and degree of degradation, and their social and economic impacts. At this level, political factors are likely to be considered and may sometimes even overrule other considerations. Information on the impact of the new management practices on each watershed, at the regional level, is used for further analysis at the national level in order to improve decision making.

- *At the regional level,* decisions mostly deal with the identification of the proper combination of management scenarios. Usually the objective at this level is to have maximum positive environmental impact on the whole underlying watershed. This combination and its corresponding impact is aggregated and quantified at the national level for approval, as mentioned above. Subcatchments are also analyzed and ranked at the regional level. The ranking is used to prioritize sites for further analysis and protection [Bishr et al., 1995]. Scenarios with the highest ranks are implemented on the local scale.
- *At the local level,* scenarios for areas with the highest priority are implemented. Their results and impacts are used to provide a feedback to the regional scale for modifying and improving the scenarios.

Providing a DSS to each level of decision making will render more reliable and properly assessed decisions. Furthermore, linking these DSS together will allow analysts to assess the impact of their decisions on the other disciplines as well as at the different levels interactively. The emphasis of the next section is on showing the data modeling and system architecture problems that may arise when linking several DSSs, to provide a multi-level decision support system, MLDSS.

2.4 Data Modeling Perspective

The investigation showed that the MLDSS should be able to support three main activities in watershed management:

- The *monitoring* of watersheds and basin status in order to keep records of the type and rate of degradation.

- The *analysis* of watersheds in order to investigate the causes of their degradation and propose plans for sustainable management.
- The *management* of watersheds, where the actual execution of the plans is performed.

These three major activities are performed by three different groups. Each group has its own database schema, as shown in Figure 2-3. At the lowest level of the schema, are the elementary objects. Elementary objects are those at the lowest or most fundamental level in a particular schema. The following briefly outlines these elementary objects:

- Land cover and land use: statistical and spatial information for agricultural, natural, urban, industrial, and agricultural areas.
- Relief: height and average slope information.
- Soil: soil mapping units of homogeneous soil characteristics and profiles with their chemical information.
- Meteorology: includes micro and macro-climate information.
- Hydrology: contains information about the hydrologic gauging, infrastructures, water quality and pollution for surface, coastal, and ground water.
- Socio-economy: includes the demographic statistics for the productive structure of the region at a municipal level. Compilation of information about the geographic characteristics of the municipalities: population, structure of the productive sectors, working population.

At the highest level of abstraction and corresponding to the three main activities of watershed management there are three different views, as shown in Figure 2-3. The views can be further abstracted to other decision levels, such as local, regional and national.

In the database schema for monitoring, elementary objects are interpreted as hydrologic response units in the context of land degradation analysis. They can be abstracted to subcatchments, catchments, and basins at the local, regional, and national levels, respectively. The effect of land degradation and the new management practices are then propagated for evaluation at the higher level.

In the database schema for analysis, elementary objects can be abstracted to various tessellations. These are the processing units for the simulation models, which are used for analyzing watersheds degradation as well as the impact of the new management practices. The tessellations can be cells, polygons, triangles, or any other geometrical forms.

In the database schema for management, elementary objects can be abstracted into management units, farms, districts, provinces, etc. Through this schema political, administrative and management constraints for land use can be propagated downwards and incorporated at the plot level. Management practices proposed in the monitoring schema will thus be transferred for implementation.

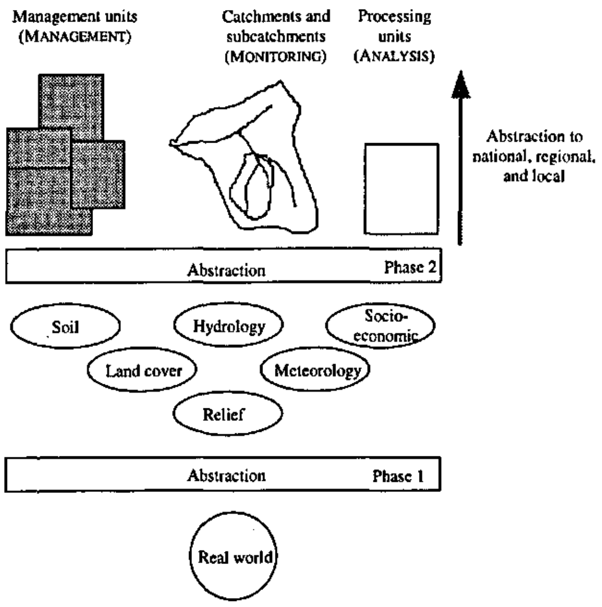

Management units (MANAGEMENT)
Catchments and subcatchments (MONITORING)
Processing units (ANALYSIS)
Abstraction to national, regional, and local
Abstraction
Phase 2
Soil
Hydrology
Socio-economic
Land cover
Meteorology
Relief
Abstraction
Phase 1
Real world

Figure 2-3 Abstraction of elementary objects to the three hierarchies.

These schemas practically reside in different databases and should be linked, in such a way that the information transfer among them is allowed at several levels. After running the simulation models, the output from the analysis schema will result in a set of new objects which, in turn, should be transferred to the spatial response units in the monitoring schema. The information on the behavior of the response units should be used to formulate land use policies in the monitoring schema. These policies should be operationalized on the management units within the management schema.

2.5 System Architecture Perspective

According to the functionality among the three levels of decision making mentioned above an experiment has been conducted, during the first stages of this study, to develop a prototype of MLDSS. During the design of the system the following criteria were considered [Bishr 1996 (b)]:

- The ability to retrieve, process, format, display, and store data, using current technology and appropriate models that will help managers, at each level, in their processes of monitoring, analysis and management of watersheds.
- The ability to provide an integrated data model which supports the various components of MLDSS.
- The ability to communicate with the different management levels by exchanging information, knowledge, and decisions.

2.5.1 Components Of the Architecture

Figure 2-4, shows the designed architecture which provides a link among several decision support systems. The architecture is based on the client-server model (more information about client-server is provided in chapter 3). It is also built on the assumption that each decision level has its own DSS. In this architecture we distinguish between two types of databases:

- The local databases, LDBs, which include the basic data required by the three levels of decision making. For this reason, a dedicated server (e.g., global server level) for abstracting and transferring these basic data to each DSS is provided.
- Each DSS has its own database. As mentioned in section 2.3, each level has its own objectives and consequently has its own view of the geographic objects in the basic databases.
- A multi-level global server is provided in order to resolve the heterogeneity among the different DSS databases.

In the following sections only the functionality of each component is emphasized, more details about the technical aspects are given in chapter 3.

Local Server (LS):

The local server represents the gateway between the local databases and the global server. The local server contains a description of the shareable information that each participating database is willing to share. The shareable information is supported by a metadatabase which contains information about all data stored in the database [Bishr 1996 (a)].

In a sense, the database schema in the LSs are abstractions of the local DB. They are responsible for accessing and retrieving information as requested by the users of the individual DSS through the Global Server. For the present work, the server hosts elementary databases of single application domain, e.g., soil, hydrology, land cover/use, relief, etc., are presented on a detailed spatial scale.

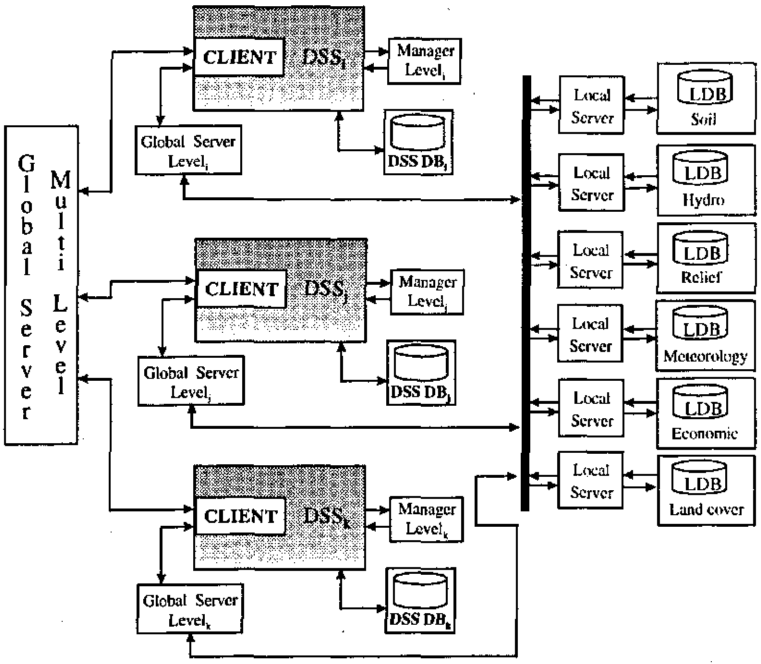


Figure 2-4 Proposed architecture of DSS.

Global Server (GS)

The global server has a federated schema, which provides a unified model for external users. To support each level, the corresponding GS links its client with the LSs. Upon receipt and acceptance of the client's request, the following operations are performed:

- Analysis of the request to identify and locate the required information and to send the corresponding messages to the appropriate LSs.
- Reception of the data sets from the LSs and the processing of these to provide the adequate information.
- Reply to the client by sending the requested information.
- Control of the transactions with the clients and data sources.
- The maintenance of the global directory, i.e., information about the data available within the federation: location, information on specific data sets, ownership, format, cost, etc. This is achieved by storing a comprehensive metadata in the global server.
- The execution of data conversion: units, formats, etc.

Multi-Level Server
In addition to its similar functionality as GS, the multi-level server is responsible for linking the different decision-making hierarchies to establish the corresponding feedback among the three decision levels in terms of information, knowledge, and decisions necessary for their activities. Its tasks are:

- The control of the communication between clients (management levels).
- Access and retrieval from the corresponding DSS database at a specific level.

2.6 The Implementation of MLDSS

The above system was partly implemented in the lab. The implementation focused on the data modeling issue. More precisely on abstracting low level elementary objects into higher level application views to support the execution of a watershed simulation model used at the regional scale. The model is cell-based and is called agriculture non-point source pollution, AGNPS [Anderson et al., 1993]. The model is embedded in a DSS where it is integrated with ARC/INFO™ and an Expert System (Nexpert Object™). Bishr and Radwan provide a detailed description of the components of such a DSS [Bishr et al., 1995]. They also present an expert-system-based integration of GIS and simulation models. The system was then extended to develop a global model. The basic objects were abstracted into a global model to support two simulation models [Espinoza, 1995; Mabote, 1995; Matin, 1996]. One was the AGNPS and the second was DUFLOW [Young et al., 1991]. AGNPS can simulate the effect of the various management practices at the local level of watershed management. DUFLOW can evaluate the combined effect of the management practices on several watersheds at the regional level.

2.7 Discussion

The basic assumption made during the experiment is that databases which contain the basic objects (soil, relief, etc.) reside in the same database management system. Notwithstanding the success of the implementation in mapping between the data models of AGNPS and DUFLOW on the one hand and the basic data stored in Arc/Info on the other hand, some problems and issues were identified after comparison with the requirements of the Spanish case.

1) System perspective

- The disciplines involved in environmental decision making are autonomous. Their authorities are assigned, even before any GIS technology is involved into these organizations, on the basis of their existing functionality and responsibility.

- It would be a systematic waste of expenses and resources, if similar data sets have to be collected and maintained. The collected data might be redundant and inconsistent, which may lead to wrong decisions.

- Environmental data are not concentrated in one database system. One organization may have many departments and divisions, each with its own authority to collect, maintain, and update its data.

- Internal users of an organization need to access the different databases for their daily work from various departments. External users need to have access as well, which might be limited according to the organizational constraints, or the purpose of use.

- Users are not willing to interact with unfamiliar interfaces, when accessing remote databases.

2) Data model perspective

- Although the AGNPS is used in the experiment, environmental analyses require different types of simulation models. Each has its own data model and input requirements. Selection of such models may depend on several criteria, two of which are the underlying problem (e.g., water quality, climate) and the level of analysis (e.g., local, regional, or national).
- Despite the client-server approach used in the design of the DSS, the provision of a global data model, at the server, is not flexible enough. This is due to the fact that the new paradigm of the client-server allows a client in one case to be a server in another. Moreover, global models cannot accommodate system evolution and autonomy requirements.
- Global schemas cannot always support external users. This is due to the fact that users still must have an understanding of the elements of the global schema.

2.8 Conclusions

From the above discussion it may be concluded that the client-server architecture is fundamental for the design of DSS. This chapter ends with a list of requirements which will be carried out as guidelines for the remaining part of this research work:

- The development of a model that allows information sharing among distributed databases is required.
- The model should maintain the autonomy of the underlying databases, while eliminating the drawbacks of the global data model approach.
- It is important to provide a mechanism which will allow users to search for relevant information resources.
- The model should accommodate the new client-server paradigm, where a client can be a data provider (server) in some cases.
- A mechanism is required by which users can retrieve data from other information resources without the need to understand the underlying

data models or interfaces. This mechanism should equally support internal and external users.

References

Anderson, M.P., Ward, D.S., Lappala, E.G., and Prockett, T.A., 1993. *"Computer Models for Subsurface Water"*. In Handbook of Hydrology (Maidment, D.R., ed.), Mc.Graw-Hill, Inc., pp. 22.1 - 22.34.

Bailey, R.G., 1996. *"Multi-Scale Ecosystem Analysis"*. In Global to Local Ecological Land Classification. (Sims, R.A., Corns, I.G.W., and Klinka, K., eds.). Kluwer Academic Publishers.

Bishr, Y., and Radwan, M., 1995. *"Preliminary Design of a Decision Support System for Watershed Management"*. ITC Journal, 1995-1. International Institute for Aerospace Survey and Earth Sciences, ITC, P.O.Box 6, 7500 AA, Enschede, the Netherlands.

Bishr, Y., 1996 (a). *"A Mechanism for Object Identification and Transfer in a Heterogeneous Distributed GIS"*. The 7th International Symposium on Spatial Data Handling, August 12-16, 1996, Delft, The Netherlands. (Kraak, M..J., and Molenaar, M., eds.)

Bishr, Y., 1996 (b). *"A Hierarchical Spatial Canonical Data Model – Towards Federating Heterogeneous GISs"*. Proceedings of the ISPRS Conference IWG III/IV- Conceptual Aspects of GIS, 1996, Vienna.

Boar, H.B., 1993. *"Implementing Client/Server Computing, A Strategic Perspective"*. McGraw-Hill, Inc. 1993. ISBN 0070062153.

Espinoza, E. J., 1995. *"Development of a Federated Database System in a Distributed Environment for Environmental Decision Making at Different Scales"*. ITC M.Sc. thesis. International Institute for Aerospace Survey and Earth Sciences, ITC, P.O.Box 6, 7500 AA, Enschede, the Netherlands.

Forman, R.T.T., and Godron M. 1986. *"Landscape Ecology"* New York, John Wiley & Sons, 619 pages. ISBN 0-471-87037-4.

Lee, R.G., Flamm, R., Turner, M.G., Bledsoe, C., Chandler, P., De Ferrary, C., Gottfried, R., Naiman, R.J., Shumaker, N., and Wear, D., 1993. *"Integrating*

Sustainable Development and Environmental Vitality. A Landscape Ecology Approach". In Watershed Management, Balancing Sustainability and Environmental Change. (Naiman, R.J., ed.). Springer-Verlag, pp. 499-521.

Mabote, T., 1995. *"The Design of a Prototype to Support Watershed Management Using The Client-Server Concept in a Distributed Environment"*. ITC M.Sc. thesis, International Institute for Aerospace Survey and Earth Sciences, ITC, P.O.Box 6, 7500 AA, Enschede, the Netherlands.

Matin, M.A., 1996. *"Development of a? Prototype for Object Identification in Heterogeneous FGIS"*. ITC M.Sc. thesis, International Institute for Aerospace Survey and Earth Sciences, ITC, P.O.Box 6, 7500 AA, Enschede, the Netherlands, 95 pages.

Risser, P.G., 1993. *"Impacts on Ecosystems of Global Environmental Changes in Pacific Northwest Watersheds"*. In Watershed Management, Balancing Sustainability and Environmental Change, (Naiman, R.J., ed.), Springer-Verlag, pp. 12-24 .

Young, A., Onstand, C. A., Bosch, D.D., and Anderson, W. P., 1991, *"Agricultural Non-Point Source Pollution Model (AGNPS)"*, User's Guide Version 3.65, July 1991, USDA-ARS, Morris, Minnesota, 56 pages.

Radwan, M., and Meijerink, A., 1993. *"The Development of a Decision Support System for Environmental Decision Making"*. Internal report, ITC Research Project (1994-1996). International Institute for Aerospace Survey and Earth Sciences, ITC, P.O.Box 6, 7500 AA, Enschede, the Netherlands.

CHAPTER 3 Interoperability of GIS

"Knowledge in the form of an informational commodity indispensable to productive power is already, and will continue to be, a major—perhaps the major—stake in the worldwide competition for power. It is conceivable that the nation-states will one day fight for control of information, just as they battled in the past for control over territory, and afterwards for control over access to and exploitation of raw materials and cheap labor."

Jean François Lyotard (b. 1924), *The Postmodern Condition: A Report on Knowledge, Introduction (1979).*

3.1 Introduction

In the previous chapter it was shown that information sharing is essential for a reliable and efficient decision-making process. The chapter concluded with a list of guidelines for the development of a mechanism for information sharing. With such guidelines in mind , we attempt in this chapter to review the current technology and research efforts to achieve this goal. As a result of this review the architecture of a proposed system for information sharing is presented. The system is called Semantic Web, SemWeb. The development of concepts, implemented in SemWeb, is the main topic of the remaining chapters of this thesis.

Section 3.2 in this chapter presents a concept known as distributed systems. The client-server model is the building block for the design of distributed systems and is presented in section 3.2.1. The notion of interoperability is introduced in section 3.2.2. Interoperability can exist at different levels. This issue is tackled in section 3.3. The overall architecture of SemWeb is presented in section 3.4. The chapter is concluded in section 3.5, where the guidelines for the research work mentioned in section 2.7 are revisited, and the main characteristics of SemWeb are considered.

3.2 Distributed Systems

The term distributed systems refers to a distributed collection of users, data , software and hardware, whose purpose is to meet some defined objectives [Brenner, 1993; Brunt et al., 1992; Deignan et al., 1993]. A comprehensive description of a distributed system design requires three levels of specification. These are physical networking, system services, and application software. The overall designed system is known as distributed computing environment or DCE.

In general computer systems provide four types of interrelated services, as shown in Figure 3-1. The data storage services provide users with efficient storage media. The data access services provide functions for retrieving data from the storage media. The application services provide users with capabilities to execute specific tasks, for example it can be a database management system, a geographic information system, etc. Finally the presentation services provide display facilities and user interfaces to end users.

There are several design possibilities for distributing any of these services. For example data can be located in distributed storage media. An application might send requests to several data access services located at different systems. A user can be provided with a single presentation service which transparently accesses different distributed application services. This particular case of distributed services is known as distributed database systems.

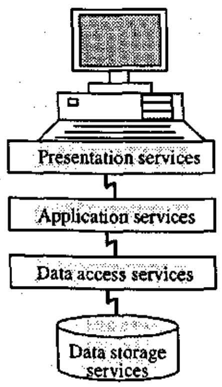


Figure 3-1 Distribution possibilities.

In the context of GII, mentioned in chapter 1, the main objective of DCE can be to provide GIS users with means to share spatial information as well as to provide application and representation services in a heterogeneous, distributed environment. The fundamental model by which the DCE is implemented is known as the client-server model. The model is summarized in the subsequent section.

3.2.1 The Client-Server Model

The client-server model is a simple model of computing, in which system functionality is divided among the components that make requests (the clients) and the components that respond to them (the servers). For example, with regard to Figure 3-1, a presentation service sends requests to the application service to perform some processing. In this case the presentation service is the client, while the application service is the server. The client and the server components are typically situated in different computers. The main concepts of the model are:

- *Server:* an application component which performs services in response to requests sent by clients.
- *Client:* an application component which sends requests to servers and receives the results of the services returned from the server.
- *Service:* could be data, analytical functions, etc., provided by the server, to the clients.
- *Client-server:* interaction consisting of one or more service request and response.

An important feature of the client-server model is that a client in some cases may become a server in other cases. This symmetrical relationship is sometimes referred to as *peer-to-peer* connection. Peer-to-peer connection forms the basic requirement for the implementation of the concepts developed in this research.

A major difficulty in the client-server model, and consequently in the DCE, is that many different standards apply. The implementation of the different standards resulted in heterogeneity among the clients and the servers. The heterogeneity might vary from the cabling system which links the clients and the servers on the one hand to the software applications on the other. It is here that the need for interoperability prevails.

3.2.2 Interoperability Defined

Interoperability is the ability of a system or components of a system to provide information sharing and inter-application cooperative process control. As shown in Figure 3-2, two systems X and Y can interoperate if X can send requests for services R to Y on the mutual understanding of R by X and Y, and Y can return responses S to X based on the mutual understanding of S, as responses to R, by X and Y.

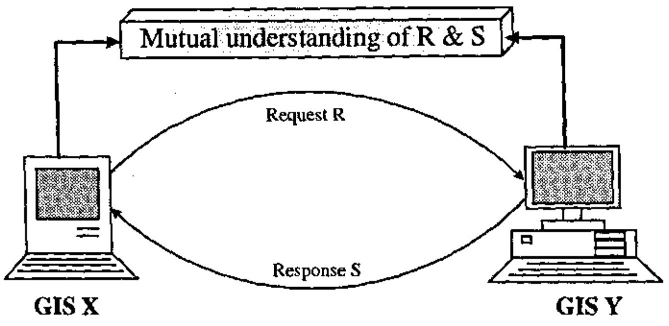


Figure 3-2 Interoperability requires mutual understanding of requests and responses.

The subsequent section provides a review of the trends and efforts of the research community and industry to provide interoperability among information systems. This helps to identify the position of this research.

3.3 Levels of Interoperability

Despite its clear definition, interoperability is mostly used to imply different things. The mutual understanding of requests and responses among systems depends on where it is applied. The notion of interoperability used by network designers, operating systems designers, and application software engineers suggests different meanings. As shown in Figure 3-3, interoperability can be viewed at six different levels, where network protocols are at the lowest level and the application interoperability, which is the focus of this research, is at the highest level.

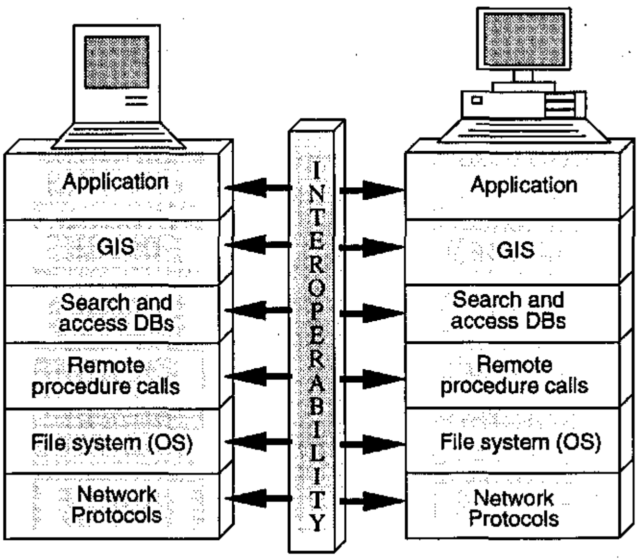


Figure 3-3 Levels of interoperability.

In the next section we will investigate each level in detail. Current trends and research efforts in each level will also be outlined. Interoperability at the lower level makes it possible to develop interoperable components at the higher levels.

3.3.1 Network Protocols Interoperability

A computer network consists of both hardware and software. The hardware includes network interface cards and cables that link them together. The software includes network protocols (e.g., TCP/IP, SPX/IPX, NETBEUI, etc.). There are several types of cabling scheme that connect computers in a network (e.g., linear, star, token ring, etc.). These schemes are known as network topology [Palmer-Stevens, 1992]. Network protocols are the rules and procedures used on a network to communicate among systems which are connected in a cable system. Protocols govern two levels of communications. High-level protocols define how applications communicate and lower-level protocols define how signals are transmitted over a cable.

The International Standards Organization, ISO, developed a seven-layer protocol known as the open system interconnection, OSI. These are the physical layer, the data link layer, the network layer, the transport layer, the session layer, the presentation layer, and the application layer. The bottom four layers form the lower level protocols and are hardware and software oriented. The upper three layers form the high-level protocols and are software oriented.

Currently there exists a wide spectrum of network protocols. Vendors no longer focus on a single protocol architecture, but instead, provide support for a variety of protocols which are known as protocol suites. The suites are packages of protocols which work according to OSI specifications. The layered architectures provide a common ground for the design of interoperable network protocol products. Examples of the suites include:

- Systems Network Architecture, SNA, by IBM.
- Digital Network Architecture, DNA, by Digital.
- NetWare, by Novell.
- AppleTalk by Apple Macintosh.
- LAN manager, which is developed jointly by IBM and Microsoft.

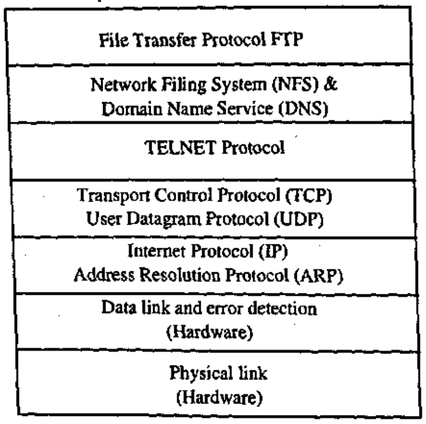

File Transfer Protocol FTP
Network Filing System (NFS) & Domain Name Service (DNS)
TELNET Protocol
Transport Control Protocol (TCP) User Datagram Protocol (UDP)
Internet Protocol (IP) Address Resolution Protocol (ARP)
Data link and error detection (Hardware)
Physical link (Hardware)

Figure 3-4 The TCP/IP suite of protocols based on OSI (after Robinson, 1993).

Perhaps, the most popular protocol is the transmission control protocol/Internet protocol, known as TCP/IP. The Internet network, which connects millions of computers around the world, relies on TCP/IP. The TCP/IP is a suite of protocols not just one protocol, as shown in Figure 3-4.

TCP/IP operates on almost any network medium, hardware, and operating system. The suite is based on providing each user with a unique address in the network [Deignan et al., 1993].

According to Shimmin, research efforts in networking are focusing on four main issues [Shimmin, 1993]. These efforts are listed below with an indication of their relevance to the GIS domain, as viewed in this thesis:

- Providing high-speed network communication hardware and protocols in order to accommodate the expanding number of network users as well as the large data sets which are characteristic of geospatial information.
- Providing interoperability among the different protocol suites and configurations. This will help to overcome the platform heterogeneity among GISs.
- Managing the numerous protocol suites. Here the focus is on providing hardware devices as well as software to support network managers in configuring and measuring the performance of the networks. This active area of research will assist in devising techniques to optimize the performance of the connected GISs.
- Providing security to the network users. As organizations build large networks, network resources, and stored geospatial information become more widely available, and become vulnerable to security threats.

3.3.2 File System Interoperability

Multiprotocol strategy only provides links at the network level. Just because a UNIX workstation can communicate with a PC using a TCP/IP protocol, does not mean that it can run applications or access files on the PC. File system interoperability allows users to open files on other systems and display them in their native formats. The interoperable file system provides this capability by extending local file system models to the network, allowing the use of files on remote machines. This is not just a matter of file transfer and access; it includes files and directories naming, access control, access methods, and file management.

The network file system, NFS, enables file sharing among systems on the network. With NFS, systems on the network are identified as clients or servers. NFS retains knowledge about the location of all accessed files in the network and all the clients that have the right to access the files. Thus, clients can issue commands to these files without having to know where the files are physically located. NFS is implemented on several operating systems to allow interoperability among their file systems, and consequently interoperability among the file systems of the GISs.

3.3.3 Remote Procedure Calls Interoperability

The network file system allows access of remote files, but it does not allow the execution of programs on another system. Remote procedure call, RPC, is a set of operations that executes procedures on remote systems. It standardizes the way programs run under the control of another operating system without having to adhere to the processing call of their underlying operating system. In this case users can run programs on remote systems independent of any operating system. One of the most common RPC mechanisms available today for development is the TI-RPC technology originally developed by Sun Microsystems.

The common object request broker CORBA and OLE/COM are considered the target interoperability technology for object-oriented DCE. More details can be found in [Kim, 1995; Common Object Request Broker, 1995].

3.3.4 Database Search and Management Interoperability

Distributed database systems may run on different hardware and operating systems which can be connected together over a high-speed network. Provided that interoperability is achieved at the lower three levels, as shown in Figure 3-3, NFS and RPC allow users to access remote files and execute remote programs respectively. This, however, is not sufficient to provide users with the ability to search and manipulate distributed databases. Distributed databases face many challenges which can fall into two categories: location of stored data and distributed data access:

1) Location of stored data: according to Robinson there does not seem to be consensus in the industry about the best or recommended data distribution strategies [Robinson, 1993]. Three strategies can be defined as follows:

- Centralization, which should not be interpreted as attempting to put all the information of an organization on one computer system. However, information which is highly correlated and usually shared can be put together. This means that an organization can have several centralized databases, each for a specific purpose.
- Distributed autonomous database schemas, which are independently designed and administered. This is the strategy adopted and implemented most.
- In the above strategy there is no direct relationship between the database schemas. However, it is also possible to partition a single schema across multiple computers, and then provide users with a mechanism to view the entire schema. Most of the academic research in distributed databases falls under this strategy [Laurini, 1994; Ozsu et al.1992; Garcia-Solaco et al., 1996; Saltor et al., 1991].

2) Distributed data access makes use of three different strategies:

- Distributed application processing allows users to send a single query which can be parsed and processed on different distributed databases. The RPC, which is mentioned above, is a viable technique to achieve this objective.
- Data access middleware: in this strategy a third-party software (middleware) provides users with transparency in accessing several distributed, independently designed, and autonomous databases [Bordie, 1992]. Middleware provides end users with a single query interface (e.g. SQL), while at the back end it can be connected to several distributed heterogeneous database management systems (e.g., Oracle, Ingress, Access). The open database connectivity, ODBC, by Microsoft, has the goal of allowing any ODBC compliant client application to interoperate with any ODBC compliant database management system.
- Data Warehouse is a synonym for clearinghouse. The technology has prevailed when the need for strategies for locating relevant information in a network of several distributed information resources has increased. A clearinghouse stores a summary of the information resources (metadata). Users can query the clearinghouse for a data set.

Once the data is located, the clearinghouse provides a mechanism which allows users to retrieve the relevant data set. Several research activities are focusing on clearinghouse developments. Alexandria, InfoHarness, GeoChange and DeltaX are some examples of these activities [Shklar et al., 1995 (a) & (b); Otoo et al, 1994; Drew et al., 1994; Frew et al., 1995].

3.3.5 GIS Interoperability

In the previous sections we did not refer to spatial databases. Up to the level of data search and management, interoperability is applicable among databases regardless of their contents (spatial or non-spatial). The lower four levels of interoperability, shown in Figure 3-3, provide a distributed computing platform where interoperable GIS can be built on. By GIS interoperability we mean that users can transparently access and share remote spatial databases and other spatial services, regardless of their underlying GIS platform.

In response to the need for interoperability at the GIS level, the Open GIS Consortium, OGC, was formed in 1994. OGC is developing software specifications known as the open geodata interoperability specification , OGIS. The OGIS framework includes three parts [Buehler et al., 1996]:

- Open geodata model (OGM), a common means for representing real world phenomena, mathematically and conceptually. It is worth mentioning that the formal data structure, FDS, is another alternative of the geodata model, which is similar in many aspects to the OGM.
- OGIS service model, a common specification model for implementing services for geodata access, management, manipulation, representation, and sharing among information communities.
- Information communities model: a framework for using the open geodata model and the OGIS services model to solve not only technical problems, but also the institutional problems.

Perhaps one of the first products which attempt to conform with OGIS requirements is the GeoMedia® product by Intergraph. With GeoMedia, users can link to distributed spatial databases running under different GIS (e.g., MGE, ARC/Info, etc.). Users can transparently access these databases with the same interface of GeoMedia.

3.3.6 Applications Interoperability

This level of interoperability is the focus of this research. The specifications provided by OGM or FDS do not prescribe the method by which users abstract and represent data. Users of geodata use GIS to build their own applications. Different applications have different world views, different representations, different schemas and hence different semantics. The technology provided by OGIS, however, is built upon the assumption that users are aware of the schemas, terms used and correct interpretations of the underlying databases. However, in a distributed environment which consists of a large number of independently developed geospatial databases, this becomes a doubtful assumption.

Heterogeneity at the application level is a semantic problem, which is due to the differences in the interpretation of the spatial data encoded in the database. We consider application interoperability and semantic interoperability as synonyms. In the real world semantic interoperability is achieved by providing metadata about the related data set (more about metadata can be found in chapter 7). Notwithstanding the ability of the metadata to provide an insight into related data sets, users are required to map the retrieved data from the domain of the provider to their own domain. The availability of semantic translators is required to support this task. Semantic translators are middleware components which allow heterogeneous applications to communicate and share data.

> A semantic translator is a middleware which can map among spatial database schemas while preserving their semantics.

The concept of semantic translators (also called mediators) was first coined by Wiederhold [Wiederhold, 1992]. It might be claimed that up to the time of the writing of this thesis, no work or research activity has been reported on semantic translators or providing interoperability at the GIS application level. Buehler and McKee, make the same observation [Buehler et al., 1996]. However, in computer science several research efforts are currently active to provide semantic translators for other business applications [Goh et al., 1994; Dumais et al., 1996].

The translator should reflect the world views of the applications benefiting from it. Hence it is rather important that the designer should have a clear understanding of the underlying applications.

3.4 SemWeb a System for Sharing Spatial Information

Up to this point we have introduced the levels of interoperability and indicated that the research focuses on providing interoperability at the application level. In this section the general architecture of the SemWeb prototype which is developed in this research is presented. SemWeb implements concepts for applications interoperability. The prototype is built on the available technologies and levels of interoperability mentioned above.

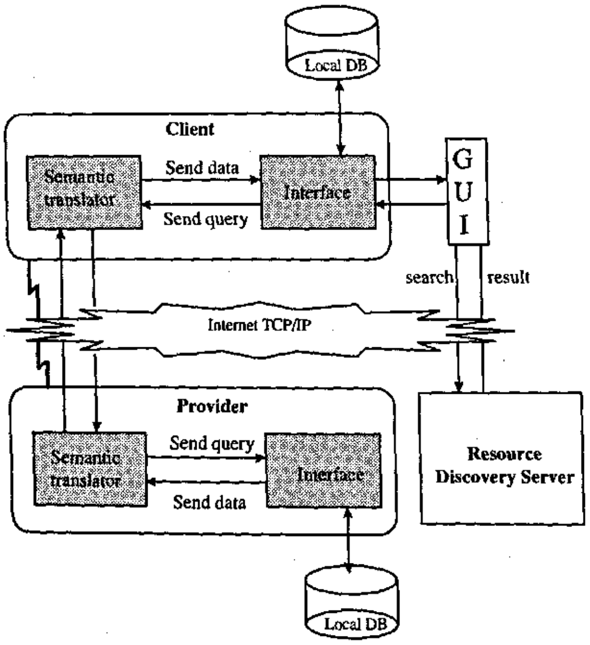


Figure 3-5 Architecture of SemWeb.

The components of SemWeb, shown in Figure 3-5, are described briefly in this chapter (the detailed implementation of SemWeb is presented in chapter 8). The relationship among the components of SemWeb and the levels of interoperability, mentioned above, are outlined. SemWeb attempts to achieve two main goals:

1) To assist users to locate an information provider in a web of information resources. This is known as the resource discovery problem.
2) To provide the users with a mechanism to share information transparently, using semantic translators

SemWeb has three main components: the client, the provider, and the resource discovery. SemWeb is implemented on top of the lower four levels of interoperability, i.e., network protocols, file system, remote procedure calls, and data management and access. The resource discovery is similar to the data warehouse mentioned in section 3.3.4. This component has a database which maintains a registry and metadata of all the available information resources. Users can search the database for relevant information resources. Due to the fact that users need to access this database across the network, the data access middleware ODBC is used. The resource discovery operation returns to the user the network address of the information resource. A model for the resource discovery is developed and presented in chapter 7. The model is called the resource discovery server, RDM.

At this stage the semantic translator can play its role in mapping between the database of the provider and the user. As can be shown in Figure 3-5, the semantic translator exists at the client and the information resource. Based on the fact that the translators can communicate together, the database designers have only to map between their databases and the semantic translator. A model for semantic translation is developed in this research and is presented in chapters 4, 5, and 6. The model is called semantic formal data structure, SFDS.

The semantic translator adopts the formal data structure, FDS, as its syntactic layer. It is worth mentioning here that it is also possible to adopt the OGM part of OGIS. However, the main reason for selecting FDS was that it has already been implemented in the lab [Pilouk, 1996; Peng, 1997; Kufoniyi, 1995]. Thus, it provides a proven theoretical ground to develop the concept of a semantic translator. The general characteristics of SemWeb are as follows:

- Users can access the database of the resource discovery server and search the metadata for relevant information resources.

- Users query the information resources, using their own terms and concepts.
- The semantic translator can map these queries from the user to the information resource.
- The semantic translator can map between the schemas of the information resource and that of the user.

3.5 Conclusions

In this chapter the meaning of interoperability is put in perspective. Six levels of interoperability are introduced. These are the network protocols, file systems, remote procedure calls, database search and management, GIS, and application interoperability. Thanks to Wiederhold the notion of semantic translators can be used as the starting point for achieving interoperability at the application level [Wiederhold, 1992]. The semantic translator provides a uniform way for passing queries and map among database schemas, back and forth, between it and the individual databases.

A general description of the SemWeb prototype, which is developed in this research, is introduced. This prototype allows users to search for relevant information resources and transparently share their information. The semantic translator forms the backbone of SemWeb.

The design of SemWeb and its underlying data models (which are explained in the subsequent chapters) attempts to maintain the guidelines mentioned in section 2.7. Installing the semantic translator on top of the existing local databases, maintains their autonomy. Database designers have only to provide mapping between their local database and the semantic translator. Providing a bi-directional mapping between the local database and the semantic translator conforms with the current trends in client/server design, where a client can also be a server.

References

Bordie, M.L., 1992. "*The Promise of Distributed Computing and The Challenge of Legacy Information Systems*". In Proceedings of the IFIP WG2.6 Database Semantics Conference on Interoperable Database Systems (DS-5), Lorne, Victoria, Australia, 16-20 November, 1992. (Hsiao, D.K., Neuhold, E.J., and Sacks-Davis, R., eds.), pp. 1-31, ISBN 0 444-898794.

Brenner, J., (ed.), 1993. *"Distributed Application Services"*. Open Framework. Prentice Hall International Ltd., Campus 400, Maylands Avenue Hemel Hempstead Hertfordshire, HP2 7EZ, United Kingdom. ISBN 0 13 630518 0.

Brunt, R., and Hutt, A., (eds.), 1992. *"The Systems Architecture: an Introduction"*. Open Framework. Prentice Hall International Ltd., Campus 400, Maylands Avenue Hemel Hempstead Hertfordshire, HP2 7EZ, United Kingdom. ISBN 0 13 560186 X.

Buehler, K., and McKee, L. (eds.), 1996. *"The OpenGIS™ Guide, Introduction to Interoperable Geoprocessing"*. Open GIS Consortium, Inc., 35 Main Street, Suite 5, Wayland, MA 01778, USA.

Common Object Request Broker (CORBA) - Architecture and Specification. Revision 2.0, July 1995, updated July 1996.

Deignan, F., and Hollingsworth, D., (eds.), 1993. *"Networking Services"*. Open Framework. Prentice Hall International Ltd., Campus 400, Maylands Avenue Hemel Hempstead Hertfordshire, HP2 7EZ, United Kingdom. ISBN 0 13 630393 5.

Drew, P., and McInnis, D., 1994. *"A Heterogeneous Geographic Information Architecture for Hong Kong Infrastructure Systems"*. IEEE Workshop on Metadata for Scientific and Technical Data Management, National Archives II, Washington, D.C., May 1994.

Dumais, S.T., Landauer, T.K., and Littman, M.L., 1996. *"Automatic Cross-Linguistic Information Retrieval Using Latent Semantic Indexing"*. In SIGIR '96 - Workshop on Cross-Linguistic Information Retrieval, August 1996, pp. 16-23.

Frew, J., Aurand, M. Buttenfield, B., Carver, P., Chang, P., Ellis, R., Fischer, C., Garner, M., Goodchild, M., Hajic, G., Larsgaard, M., Park, K., Probert, M., Smith, T., and Zheng, Q., 1995. *"The Alexandria Rapid Prototype: Building a Digital Library for Spatial Information"*. A Forum on Research and Technology Advances in Digital Libraries, pp. 173-195, McLean, VA., May, 1995.

García-Solaco, M., Saltor, F., and Castellanos, M., 1996. *"Semantic Heterogeneity in Multidatabase Systems"*. Object-Oriented Multidatabase Systems, (Bukhres, A.O., and Elmagarmid, K.A., eds.). Prentice Hall, Englewood Cliffs, NJ 07632, pp. 129-202. ISBN 0 13 103813 3.

Goh, H.C., Madnick, S., and Siegel, M., 1994. *"Context Interchange: Overcoming the Challenges of Large-Scale Interoperable Database Systems in a Dynamic Environment"*. In Proceedings of the Third International Conference on Information and Knowledge Management, Gaithersburg, MD, November 1994, pp. 337-346.

Kim, W., 1995. *"Modern Database Systems, The Object Model, Interoperability, and Beyond"*. ACM Press, A Division of the Association for Computing Machinery, Inc., Addison-Wesely Publishing, ISBN 0 201 59098 0.

Kufoniyi, O., 1995. *"Spatial Coincidence Modelling, Automated Database Updating and Data Consistency in Vector GIS"*. A Ph.D. Thesis, Wageningen Agricultural University, The Netherlands, 206 pages.

Laurini, R., 1994. *"Sharing Geographic Information In Distributed Databases"*. Proceedings URISA, 1994, pp. 441-454.

Otoo, J.E., and Mamhikoff, A., 1994. *"Delta-X Federated Spatial Information Management System"*. Proceedings of ISPRS commission II Symposium on System for Data Processing Analysis and Presentation, Ottawa, Canada. (Allam, M., and Plunkett, G., eds.). Vol. 30, No. 2. The survey, mapping and remote sensing sector, natural resource Canada.

Ozsu, M.T., and Valduriez, P., 1990. *"Principles of Distributed Database Systems"*. Prentice Hall International Inc., London, 562 pages. ISBN 0 13 715681 2.

Palmer-Stevens, D., 1992. *"The Cabletron Systems Guide to Local Area Networking"*. (Atkinson, S., ed.) Cabletron Systems Benelux B.V., Korenmolenlaan 1A, 3447 GG Woerden, The Netherlands.

Peng, W., 1997. *"Automated Generalization in GIS"*. Ph.D. Thesis, Wageningen Agricultural University, The Netherlands, 188 pages. ISBN 90 6164 1349.

Pilouk, M., 1996. *"Integrated Modelling for 3D GIS"*. Ph.D. Thesis, Wageningen Agricultural University, 200 pages. ISBN 90 6164 122 5.

Robinson, I., (ed.), 1993. *"Distributed Information Access Technologies"*. In Lan Times, Guide to Interoperability, Network Interconnectivity Solutions. (Sheldon, T., ed.). Osborne McGraw-Hill, 2600 Tenth Street, Berkeley, California 94710, U.S.A., pp. 261-276. ISBN 0 07 882043 X.

Saltor, F. , Castellanos, M.G., and García-Solaco, M., 1991. *"Suitability of Data Models as Canonical Models for Federated Databases"*. SIGMOD Record, Vol. 20, No. 4, December 1991, pp. 44-48.

Shimmin, F.B. (ed.), 1993. *"Network Technologies for the '90s"*. In Lan Times, Guide to Interoperability, Network Interconnectivity Solutions. (Sheldon, T., ed.). Osborne McGraw-Hill, 2600 Tenth Street, Berkeley, California 94710, U.S.A., pp. 21-45. ISBN 0 07 882043 X.

Shklar, L, Shah, K., and Basu, C., 1995 (a). *"Putting Legacy Data on the Web: A Repository Definition Language"*. Special Issue of "ISDN and Computer Networks", Proceedings of the Third International WWW Conference '95, Vol. 27, No. 6, April 1995.

Shklar, L., Sheth, A., Kashyap, V., and Shah, K., 1995 (b). *"InfoHarness: Use of Automatically Generated Metadata for Search and Retrieval of Heterogeneous Information"*. Proceedings of CAiSE '95, June 12-16, Jyvaskyla, Finland, Springer-Verlag Lecture Notes in Computer Science, Nr. 932.

Wiederhold, G., 1992. *"Mediators in the Architecture of Future Information Systems"*. Computer, March 1992, pp. 38-49.

CHAPTER

4 Abstraction and Heterogeneity Among Contexts

"For a large class of cases—though not for all—in which we employ the word "meaning" it can be defined thus: the meaning of a word is its use in the language."

Ludwig Wittgenstein (1889–1951),
Philosophical Investigations, pt. 1, sct. 43 (1953).

4.1 Introduction

In chapter 3, the concept of interoperability was explained. It was mentioned that the problem of heterogeneity in a distributed environment has two distinct and yet complementary perspectives: data modeling and system architecture. The architecture and components of SemWeb, the prototype developed in this work, were introduced in brief. This chapter, as well as the next two chapters, demonstrates several data modeling problems and a proposed solution to resolve heterogeneity amongst distributed spatial databases. This chapter explains in detail the causes and types of heterogeneity. The proposed solution is introduced in chapters 5 and 6.

The problems of heterogeneity, from the data modeling perspective, prevail when the underlying schemes of two or more independently designed geographic information systems are compared. Two communities are involved, where the problem of heterogeneity is concerned. The first community is that of the database designers or integrators, who are responsible for reconciling and federating heterogeneous, spatial databases. They have to understand the assumptions and concepts of the underlying databases. The second community is the user community, which ideally will have transparent access to the federated databases. The transparent access allows users to request and retrieve data, based on their own semantics.

The main objective of this chapter is to introduce the types of heterogeneity that might exist among geographic information systems, and to present the first step towards the proposed solution. The process of abstracting real world features to a computer presentation is described in sections 4.2 and 4.3. On the basis of the analysis of the abstraction process, the types of heterogeneity are enumerated in section 4.4. Before informally presenting the developed solution concept in section 4.6, I present a review of the current approaches and research efforts to resolve heterogeneity in section 4.5.

4.2 Abstraction of Real World Features

Fang, Hammer, and McLeod classified heterogeneity among independent databases, in analogy to the phases of database design, as conceptual, logical, and physical [Fang et al., 1991]. This classification ignores semantics as a type of heterogeneity. Semantic heterogeneity can become apparent, if the classification is based on the structural and behavioral differences among database models. Such a fact has been recognized by other authors, where they introduced semantics as a type of heterogeneity [Castellanos et al., 1991; Urban et al., 1991; Ventrone et al., 1991]. However, the authors had different a interpretation of the notion of semantics. Before a solution for heterogeneity is sought, it is then essential to give a proper definition of heterogeneity in general and semantic as well as schematic heterogeneity in particular.

The process of database design proceeds from the real world to the computer world, within a certain context. The set of features in the real world which are of interest to a particular context forms its universe discourse (UoD). The context world view is the conceptual model of the underlying UoD. The context world view is, then, an intensional definition of the UoD. It acts as a filter between the UoD and the computer representation, as shown in Figure 4-1. The definition of the context world view includes three types of abstraction. These are the categories definition, class intension definition and geometric description. These definitions are used to survey the UoD and represent it in the database. Therefore, the computer representation is an intensional as well as an extensional representation of the UoD of the underlying context. The computer representation deals with class hierarchies, objects and geometric primitives of the features abstracted from the UoD. In the following text the notion of semantics refers to the relationship between the context world view and the computer representation, unless otherwise stated.

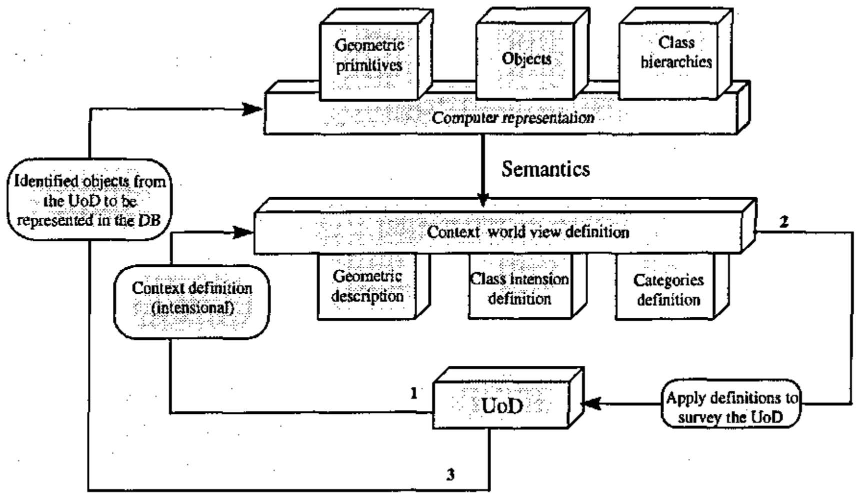

Geometric primitives
Objects
Class hierarchies
Computer representation
Semantics
Identified objects from the UoD to be represented in the DB
Context world view definition
2
Context definition (intensional)
Geometric description
Class intension definition
Categories definition
1
UoD
Apply definitions to survey the UoD
3

Figure 4-1 Abstraction of real world features: 1) define classes, rules for object/class membership, and geometric description; 2) apply these rules to survey the real world; 3) representation in the GIS domain.

Context is the domain where the process of abstracting the UoD to the computer representation occurs.

The context world view is the conceptual model of the UoD of the underlying discipline. It includes: categories definition, class intension definition, and geometric description.

The computer representation consists of an intensional and extensional representation of the UoD. It includes, class hierarchies, features and geometric representation.

Semantics is the relationship among the computer representation and the context world view within a certain context.

It is essential to distinguish between semantics, as defined above, and computer models' semantics as defined in computer science. In the latter, semantics is the mathematical interpretation of the formal language expressions of the database [Leeuwen, 1990]. Furthermore, we distinguish between a discipline and a context as follows:

- Different disciplines recognize different UoDs. For example Soil scientists observe the process of soil erosion in a watershed, while land-use planners observe suitability of soil units for different use types of the same area.
- A context defines the whole process of abstraction, as defined above, from the UoD to the computer representation. Hence, a context encompasses the context world view and the computer representation, while a discipline can be represented by several contexts.

It is assumed that for a particular area which is represented in a database, a difference in contexts implies a difference in the context world view and consequently a difference in the computer representation. The objective of the next section is to highlight these differences. This will help to define the aspects of heterogeneity that might exist among different contexts.

4.3 Types of Abstraction

To illustrate and analyze the abstraction processes, shown in Figure 4-1, an example will be presented. Within a river basin area, a hypothetical basin management project is in progress. Several disciplines may be involved in such a project: a soil and water conservation group, soil mapping group, forestry group, land-use planning group, etc. Each of these disciplines abstracts real world features and represents them in its computer model, using the definitions of its context world view. The objective of the subsequent sections is to define the types of heterogeneity that might exist between two independent GIS contexts, as a first step towards achieving interoperability.

4.3.1 Categories Definition

The process starts by realizing the concept of space, i.e., fields (e.g., areas of similar soil characteristics) or objects (e.g., rivers) which are categorized on the basis of their characteristics. The relationship between the real world and the categories definition is shown in Figure 4-2. World features are categorized differently, depending on the objective of the underlying context. In the soil mapping context, categories of homogeneous soil complexes are of interest. On the other hand, one of the objectives of the land-use management context might be to maximize land productivity at farm level. The underlying area of study is categorized into homogeneous land-use types. Detailed soil information is not usually required by this discipline.

```
Categories are collections of real world features
with similar characteristics.
```

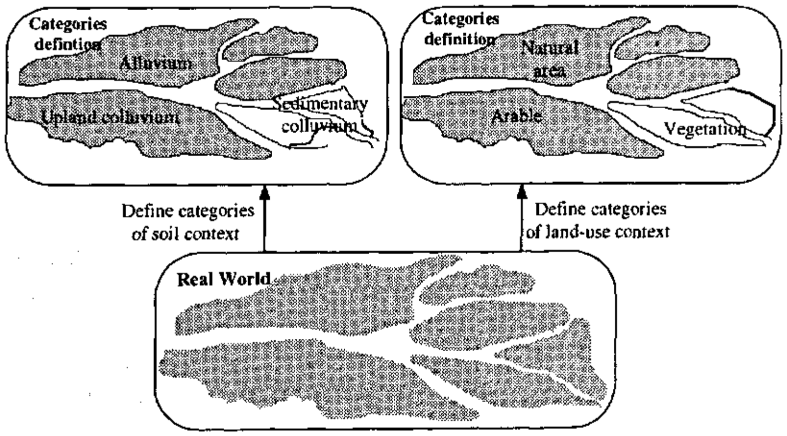


Figure 4-2 Definition of categories relevant to land use and soil disciplines.

4.3.2 Class Intension Definition

The class intension definition pertains to defining rules which are used to associate objects with categories. It is important to bear in mind that the context world view is an intensional definition and hence no actual instances are captured, only rules and possible values of the attributes (i.e., their domains) are defined. As shown in Figure 4-3, rules are defined to associate instances with specific agriculture and soil mapping units.

```
Class intension definition is the process of
defining rules by which real world features are
identified and associated with categories.
```

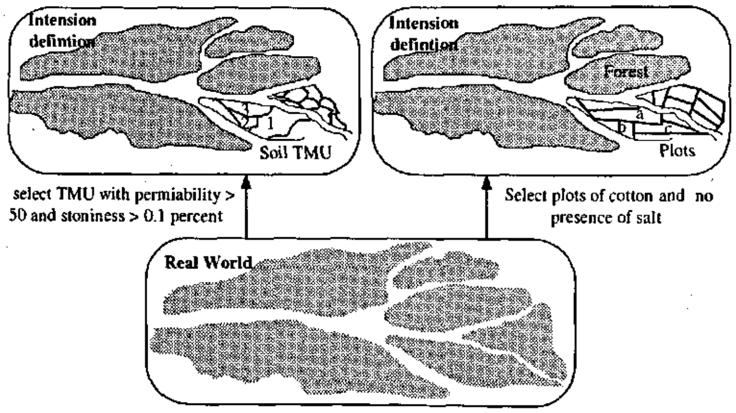

Intension definition
Soil TMU
1
Intension definition
Forest
a
b
c
Plots
select TMU with permiability > 50 and stoniness > 0.1 percent
Select plots of cotton and no presence of salt
Real World

Figure 4-3 Depending on the intension of classes, objects can be assigned to different categories.

4.3.3 Geometric Description

To perform spatial analysis it is necessary to have geometric and topologic information of relevant real world features. The geometric description is meant to provide methods and rules to assign geometric types to instances. For example, as shown in Figure 4-4, some elements of the spatial objects are of interest to the underlying context world view. Rivers are represented by their centerline in the land-use group. However, in the soil mapping group they are presented as area features.

Dimensions are also introduced in the geometric description. Dimensions which are recognized in this type of abstraction might include unary relations (such as length of an arc and bearing of a horizontal line) and binary relations (such as distance between two points, buffer zone, etc.). The resolution of the representation of spatial objects in the computer is selected. Topologic relationships are also described (such as neighborhood, containment, touch, overlap, etc.) [Molenaar, 1995]. Measurements of the spatial extent of features in the *Geometric Description* would not be possible without identifying the nodes of such features. Nodes can be represented as n-tuples, where n is usually equal to 2 or 3 (X, Y) or (X, Y, Z). The set of nodes is defined to be a collection of points sufficient for the geometric and the topologic construction of the geospatial extent of any feature of interest.

At the geometric description also a decision has to be made on whether real world features can be mapped as discrete spatial elements (e.g., vector) or as field functions that take a value at any position in a two dimensional space (e.g., raster) [Molenaar, 1989]. There are many real world features which could have aspects of both vectors and fields. For example, a stream reach could be a unique entity represented as a string of arcs, but it could also be recognized as a variable flow rate or variable water quality index from one point to another represented as fields.

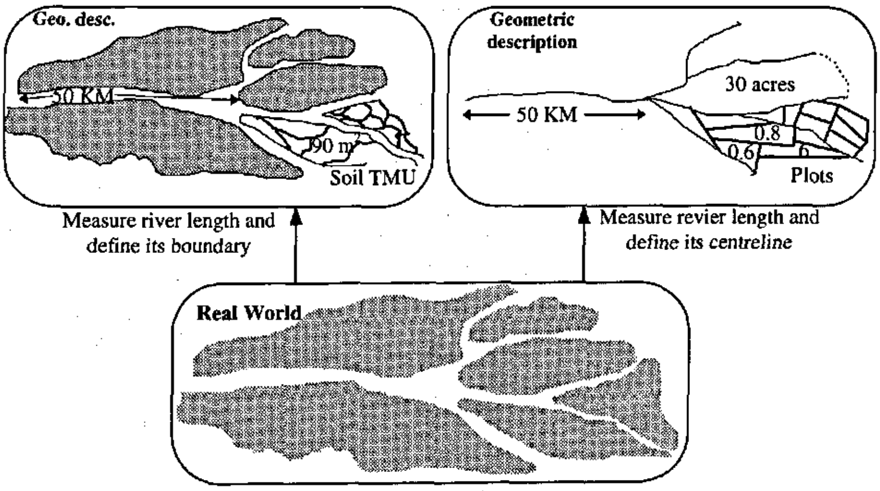

Geo. desc.
50 KM
90 m²
Soil TMU
Geometric description
30 acres
50 KM
0.8
0.6
6
Plots
Measure river length and define its boundary
Measure revier length and define its centreline
Real World

Figure 4-4 Definition of specifications to capture the geometry of real world features.

The spatial reference system and quality parameters must be defined in this type of abstraction. The n-tuples are measured and referenced to a selected spatial reference system that can be a horizontal, vertical, or linear (e.g., distance pole on a highway) datum. The spatial reference system should be maintained in the *computer representation* in order to locate features from the computer representation correctly to their original locations in the real world.

> Geometric description is the process of outlining specifications to assign geometric types to features as well as specifications to represent them in the computer. The specifications depend on the objective and the type of analysis required by the underlying context.

4.3.4 Class Hierarchy

In section 4.3.1, the notion of categories was introduced and defined. These categories are mapped to a *class hierarchy* in the *computer representation* to form the database schema. Classes have attributes structure and they may be considered as containers for objects in the database.

```
Classes are computer representations of the
categories which were defined in the context world
view.
```

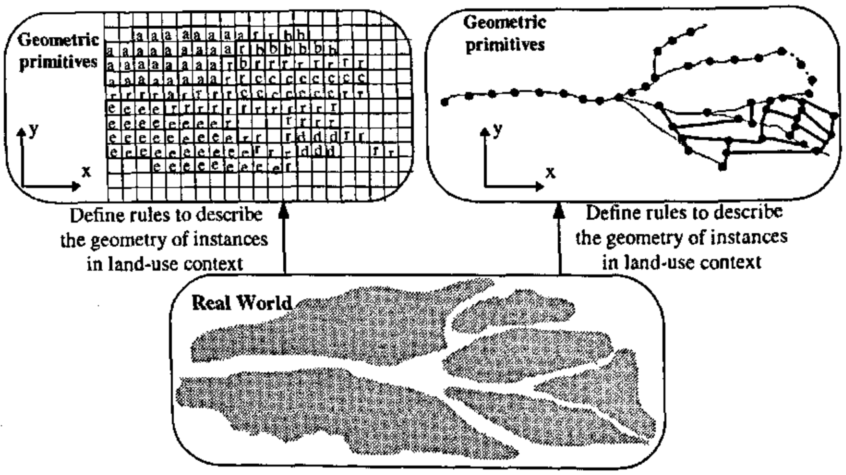

Geometric primitives
y
x
Geometric primitives
y
x
Define rules to describe the geometry of instances in land-use context
Define rules to describe the geometry of instances in land-use context
Real World

Figure 4-5 On the basis of the predefined geospatial specifications, features are presented as raster or in a planar graph geometry.

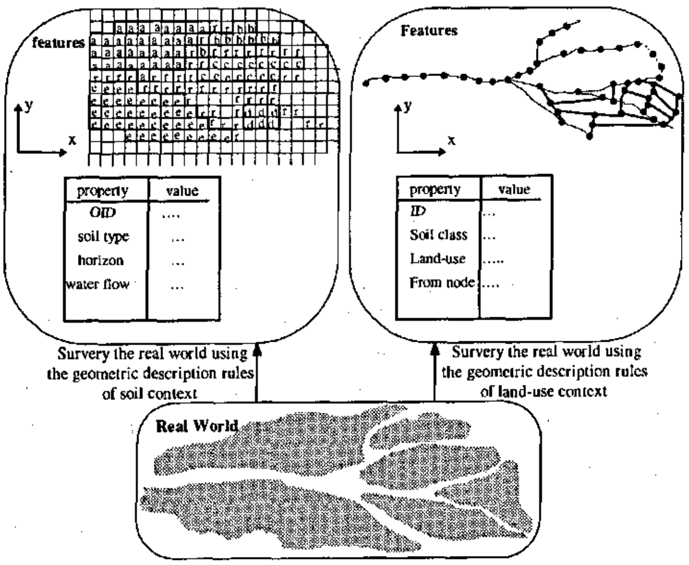
4.3.5 Geometric Primitives

Geometric description specified at the *context world view*, section 4.3.3, are used to survey real world features and represent these as objects in the *computer representation.* The geometry of objects is described by geometric primitives, nodes, edges, faces, or raster (or any other geometric element). The geometric primitives are used to construct the topologic relationships amongst the spatial objects in the database. Usually the topology is defined in a two dimensional, 2D, planar graph. However, some spatial occurrences cannot be described in 2D topology. For example two roads which cross each other at two different levels. In this situation 3D topological relationship is required. Pilouk presents an integrated model of planar graph geometry and digital terrain models [Pilouk, 1996]. As shown in Figure 4-5, real world features are represented as raster in the context of soil mapping, while they are presented as vector structures in the context of land-use.

```
Geometric primitives in the computer representation
are basic geometric elements which describe the
geometry of spatial objects. They can be described
by vector geometry (nodes, edges, faces), or raster
geometry (or any other tessellation).
```

4.3.6 Objects

Real world features are represented as objects or attributes in the database. The objects are given attribute/value pairs. Values are assigned to the attributes by a surveying process. As shown in Figure 4-6, the river basin, TMUs, and land-use plots are assigned attribute/value pairs.' The geometry of the objects is described by a collection of geometric primitives. For example the geometry of a river can be described as a chain of nodes and edges.


features
y
x
Features
y
x
Survery the real world using the geometric description rules of soil context
Survery the real world using the geometric description rules of land-use context
Real World

property	value
OID	....
soil type	...
horizon	...
water flow	...

property	value
ID	...
Soil class	...
Land-use	.....
From node	....

Figure 4-6 Geometric primitives are aggregated to form objects, *which are assigned attribute/value pairs.*

> Objects are the actual instances of the database classes. They have geometric and thematic descriptors as well as topologic relationships. The geometry can be described in a 2D planar graph, or in 3D, while the thematic descriptors are presented as a list of attribute/value pairs.

4.4 The Problem of Information Sharing

Several types of abstraction were presented in the previous section. Based on these, the types of heterogeneity among database schemes are defined in this section. The most fundamental principle on which modeling rests is a one-to-one correspondence between real world objects and their symbolic representations in the database environment. Kent mentions that this assumption is strongest in object-based systems and weaker in value-based systems (e.g., relational databases) [Kent, 1991]. The assumption starts to break down, when we deal with several databases. We might have different objects, or attributes, in independent databases representing the same real world feature. The fundamental problem of sharing databases is to maintain this consistency, when objects move among databases.

> Let x and y be two objects in two independent databases.
> We say that $x = y$ if they are the same.
> We say that $x \equiv y$ if they refer to the same real world feature.
> We can assert that the first implies the second, but not vice versa.

When we speak of interoperable geographic information systems we mean that systems should be independent and yet can transparently communicate at a high level of semantics [Bishr et al., 1996]. Heterogeneity mainly arises, when we attempt to interoperate spatial information systems. From the definitions mentioned above, it may be concluded that differences in the context world view can be described as:

- Differences in the class hierarchies and the attribute structure.
- Differences in the rules that assign objects to classes.
- Differences in the geometric representation of the spatial objects.

The above three types of differences lead to three different types of heterogeneity. These are schematic, syntactic, and semantic. Within a certain context, people categorize real world features and represent them as classes which have attribute structure and arrange them in hierarchies (schemata). They capture features and represent these as objects with thematic and geometric descriptors in the databases (syntax). During their use of the database they attempt to relate database objects to the context world view (semantics).

4.4.1 Semantic Heterogeneity

Semantic heterogeneity is usually the source of most information sharing problems. Individuals from different contexts, who share their data, are likely to share interest in a common UoD. For example, a soil mapping and conservation group is interested in defining detailed classification of TMUs, their profiles, horizon, and rate of erosion. This differs from the way land-use group recognizes features. They identify natural vegetation areas, types of land-use at farm and plot levels, soil suitability for agricultural purposes, and irrigation schemes. The land-use group regularly requires soil information, which can be retrieved from the soil conservation group. Soil information can, then, be considered as the common UoD.

It may be concluded that semantic heterogeneity occurs mainly due to differences in the context world view. In other words, due to differences in the definition of categories, differences in the definition of the intension of classes and differences in the geometric description. This set of definitions is collectively called context information. Intuitively, differences in semantics among contexts, lead to syntactic and schematic differences.

- Context information is the collection of category definition, class intension definition, and geometric descriptions
- Semantic heterogeneity occurs due to differences in context information.

4.4.2 Syntactic Heterogeneity

The different representations of real world features as fields or as objects in the database domain are directly related to semantic reference. There are two principal structures for representing spatial objects and linking their thematic and geometric data (more details can be found in [Molenaar, 1995; Molenaar, 1996]):

1) *Raster data structure:* a collection of points or cells distributed in a regular grid. Each cell in a grid is assigned to a thematic value which refers to a feature in the real world. Raster can be either single-valued, or multi-valued. In the single-valued type, the raster has one attribute representing one particular thematic aspect of the terrain. In multi-valued raster, each raster has several attributes for the same terrain segment. The topology of the raster structure is based on the adjacency of the raster point or cell.
2) *Vector data structure:* the geometry of objects can be a collection of nodes, edges, or faces (in case of planar graph geometry). The link between the thematic data and the geometric data is made through an object identifier.

A proper solution to the syntactic heterogeneity problem would be to provide a common syntax for spatial objects representation to all GISs. The open geo-information specifications, OGIS [Buehler, 1996], and the formal data structure, FDS [Molenaar, 1995], attempt to provide a formal way of representing real world features in spatial databases. However, a common syntax for object representation will still leave the problem of having the same objects presented in different databases with different geometry, as well as topologic relationships, unresolved. In this sense the syntactic heterogeneity can be classified as follows:

1) Vector geometry
   - Area Vs Area
   - Arc Vs Arc
   - Point Vs Point
   - Area Vs Arc
   - Area Vs Point
2) Raster Vs Raster
   - Mapping function
3) Raster Vs Vector
4) Spatial reference system

5) Units of measurement

6) Quality parameters

> Syntactic heterogeneity is the difference in the thematic and the geometric representation as well as the topologic relationships of spatial objects.

4.4.3 Schematic Heterogeneity

The schemata, i.e., the classes, attributes and their relationships can vary within or across contexts. For example, several classification methods exist for ecological land classification [Sims et al., 1996], also there are several methods to classify soil units ASTM and SOTER classification methods [Engelen et al., 1995; ASTM, 1985]. Moreover, two databases might adopt the same classification method and have different class and attribute structures, which results in the schematic heterogeneity. Schematic heterogeneity can be classified as follows (only a list is provided here, more details are given in chapter 6):

1) Entity type Vs Entity type
   - synonyms and homonyms
   - 1:M and M:N relationships
   - missing attributes
   - missing but implicit attributes
   - entity constraints
   - methods

2) Attribute Vs Attribute
   - synonyms and homonyms
   - 1:M relationship
   - data type
   - methods
   - default values
   - domain

3) Entity type Vs Attribute

4) Different Representation for Equivalent Data

- Different units of measurements
- Different spatial resolutions
- Different quality parameters

`Schematic heterogeneity is the difference in the class hierarchies and attribute structure of two independent database schemas.`

4.5 Mapping Between Schemes

Existing geographic information systems do not represent semantics explicitly. Usually semantics are mentally maintained by the users of a system. With regard to Figure 4-1, this situation can be accommodated in a single GIS environment. However, when data flow between systems, the relationship between the context world view and the computer representation (i.e., semantics) ceases to exist. Receivers of the data set are not informed about the details of the context world view of the providers. What they have is their own context world view, their own computer representation and hence, their own semantics.

Since we deal with database schemas it is necessary to resolve the schematic heterogeneity among the database of the receivers and that of the providers. This can be achieved by mapping between the schemes involved. Ideally, a successful mapping leaves each user to maintain the relationship between his computer model and the context world view, as shown in Figure 4-7. In the next section we shall show the approaches which were reported in the literature to resolve heterogeneity among database schemas. The first approach requires from users that they have an understanding of the schemes of the remote databases which they interact with. The second approach provides a mechanism by which users can formulate queries as well as retrieve data based on a common schema. In the third approach users can formulate their queries based on their own vocabulary and a context mediator can handle the differences from the information resource.

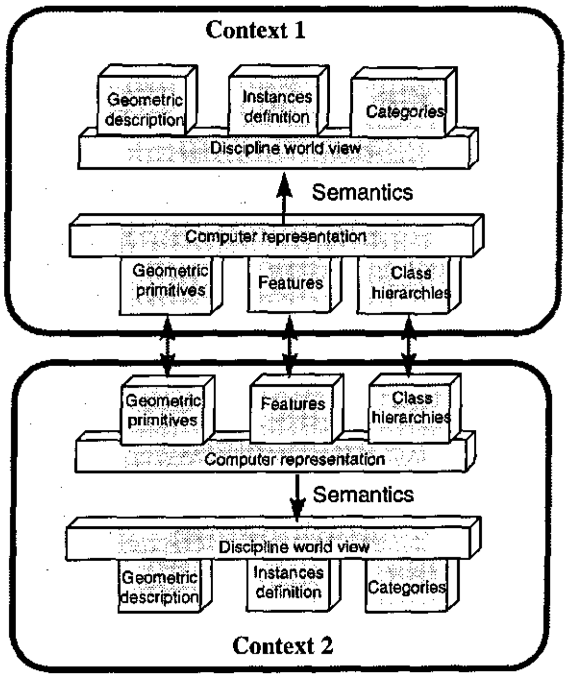


Figure 4-7 Semantics at the two contexts are maintained after mapping between their schemes.

4.5.1 No Shared Schema, No Context Mediation

This is also known as the multidatabase system, where users formulate queries using the export schemes of the information provider [March, 1990]. An export schema is a subset of the database which users are willing to share among themselves. Users take the responsibility to detect and resolve conflicts that may exist between their local schema and the export schema of the information provider.

Although this approach might provide users with a greater flexibility and full control of the system, it is not practical to overload them with the burden of acquiring full knowledge about the export schemes from which they frequently retrieve data. As shown in Figure 4-8, users at DB1, directly interact with DB2 through its export schema. Queries, submitted by a user at DB1, have to be formulated in such a way that they can be processed directly by a user at DB2. The retrieved data set is in the form of the export schema of DB2 and users then have to transform it into DB1.

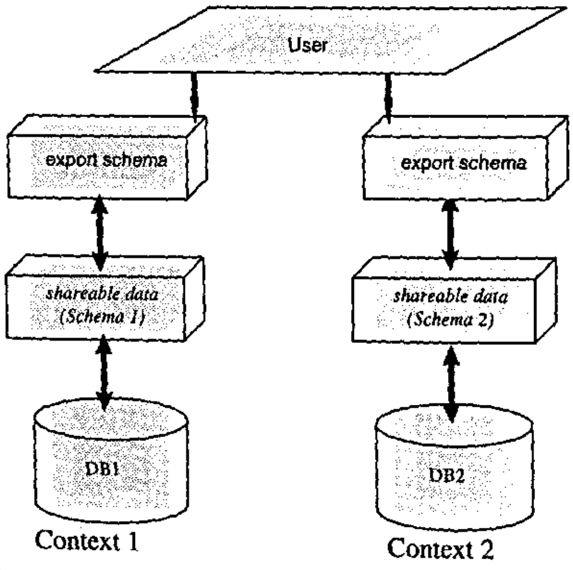


Figure 4-8 No shared schema and no context mediation.

4.5.2 Shared Schema, No Context Mediation

This approach has been widely reported in the literature [Landers et al., 1982; Arens et al., 1992; Ahmed et al., 1991]. Database designers attempt to reconcile all the conflicts among all component databases, by designing a federated schema. The federated schema (also called unified or global schema) is maintained in a server called the federation server. The server has a directory of all data sources. The system allows users to send queries based on the federated schema, as shown in Figure 4-9. The federated schema is defined as a view of the export schemes of the component databases. The definition of a federated schema incorporates functions to resolve discrepancies and inconsistencies among the export schemes of the underlying databases. The retrieved information in this approach is based on the federated schema.

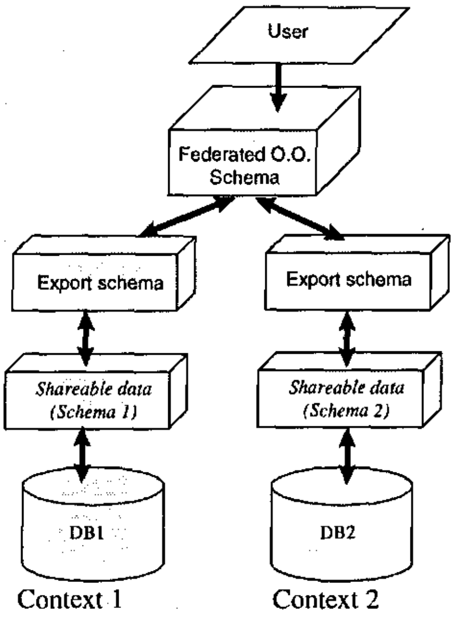


Figure 4-9 Shared schema and no context mediation.

The disadvantage of this approach is that a federated schema is not necessarily free from conflict with other export schemes, it is merely a compromise. Furthermore, it is impractical to provide a federated schema of a large number of component databases. Another problem with this approach is that the system hides from users the source of the data set which might be important in some cases to inform them about the reliability and usability of the retrieved data set.

4.5.3 No Shared Schema, with Context Mediation

In this approach users have the flexibility of formulating their queries, using their own vocabulary without the need to identify the conflicts explicitly [Collet et al., 1991]. The context mediator then handles the differences in the users' and the information resource contexts, Figure 4-10. A context mediator does a number of things each time it receives a query referencing multiple data sources. First it compares the context of the query sender with the context of the receiver, and reformulates the query in such a way that it is understood by the receiving context. This setup requires from the receiver and the sender that they establish a mapping between the context mediator and their own contexts.

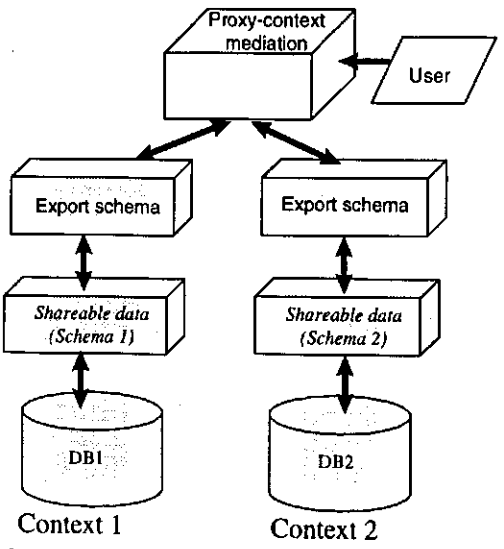


Figure 4-10 Context mediator and no shared schema.

Although the contextual problem is partially resolved in this approach, the schematic problem remains unresolved. The system only resolves the differences in naming conventions, units, spatial reference system, etc., and assumes that there is a one-to-one mapping between the export schemes of the component databases. The GeoMedia® product which was mentioned in chapter 3 is an implementation of this approach.

4.5.4 Discussion

The multi-database approach provides an explicit selection of the information resource, while the federated database approach hides the information resource from users. The context mediation approach explicitly represents and accesses the underlying context information. However, it does not provide schema mapping, because no federated schema is defined. The federated database provides a fixed link between the users and the providers in such a way that users do not have the option of explicit source selection.

It is possible to provide a direct mapping between the schemes of two databases, as shown in Figure 4-7. Although this is possible in a limited number of databases, the direct mapping becomes impractical when a large number of databases is involved. We then require an intermediate context which databases can subscribe to, if they need to share their data. If a context is to exchange information with another context, it means that they will have to share the same UoD, or at least part of it, i.e., have a common UoD. In this case it is necessary to develop a proxy context which is an abstraction of the common UoD.

4.6 The Approach Adopted in the Research

The adopted approach is called the semantic formal data structure, SFDS. It is an extension of the formal data structure, FDS. SFDS adopts a combination of the features of the federated schema and the context mediation approaches to resolve the schematic and the semantic problem. In SFDS users can send queries using their own vocabulary which are in turn transformed into the shared context and then transferred to the export schema of the information source, as shown in Figure 4-11. The data set is retrieved by transforming it to the federated schema and then to the export schema of the user. The context mediator is used to map between the schemes involved. Table 4-1 provides a comparative analysis of SFDS and the three approaches mentioned above.

SFDS is a formalism of the Proxy Context which is a mediator for sharing information among two or more GISs. The proxy context represents a common UoD, which is the domain where two or more independent databases share their information, see Figure 4-12. Information which is exchanged between any two contexts is first mapped from the sending context to the proxy context and then from the proxy context to the receiving context. Information mapped from a context to a proxy context is not necessarily mapped back to the original context without any information loss (this issue is discussed in more details in section 6.6).

```
Let U = {$U_1$, …, $U_n$} a set of UoDs,
Let Cont = { $Cont_1$, …, $Cont_n$} be contexts of U
We define $U_c$ so that
$U_c = U_1 \cap \dots \cap U_n$
A proxy context $Cont_p$ is an abstraction
from $Cont_1 \cap \dots \cap Cont_n$
```

SFDS consists of three layers. These are the syntactic layer, schematic layer and semantic layer, which are intended to resolve the syntactic, schematic, and semantic heterogeneity, respectively. The layers are explained in detail in chapter 5.

The federated schema in SFDS is different from the one introduced in federated databases. In the latter case, the schema is a global view of the export schemes of all the component databases in the federation. In SFDS, however, the federated schema supports a certain application domain. For example, one federated schema to exchange road information, another for height information, and another for soil information.

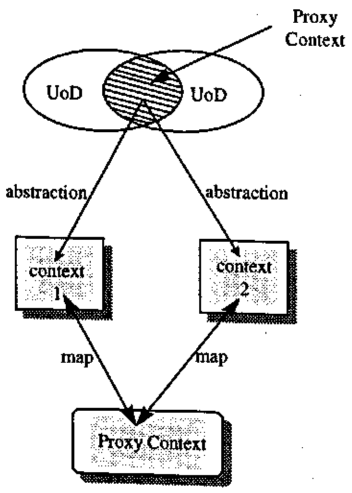


Figure 4-11 The definition of a common proxy-context..

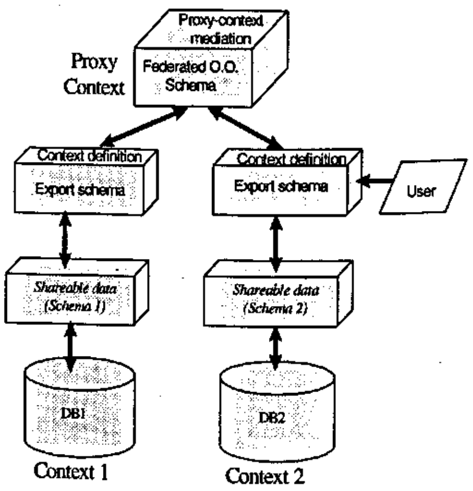


Figure 4-12 Shared schema and context mediation.

This distinction between the federated schema in SFDS and that in the federated database system is due to the fact that the former supports loosely coupled federation, which is contrasted with tightly coupled federation supported by the latter. In tightly coupled federation, database administrators have the responsibility to create and maintain a global schema for the federation and to control the access to the component databases actively [Litwin et al., 1986]. On the other hand, in loosely coupled federation, no global schema is required and there is no control enforced by the federated system and its administrators [Sheth et al., 1990]. According to the authors mentioned earlier, loosely coupled federation is a synonym for interoperable database systems, which clearly conforms with the main objective of SFDS.

Several aspects that need to be considered, while designing federated schemes, are identified:

1) *Completeness and correctness*: the federated schema should contain all possible concepts in the common UoD it represents.

2) *Clarity and simplicity*: the federated schema should be easy to understand for database designers who create the mapping between the federated schema and their export schemas. The federated schema should also be simple, so that mapping operations are easy to implement.

3) *Minimum loss of information*: class hierarchies, and attribute structures, as well as geometric representations should be designed so that information loss is minimized during the mapping process.

4) *Extendibility*: the federated schema should be designed to anticipate any possible extension or update to the schema.

Before we proceed with our explanations of SFDS in chapter 5, the following is meant to define terms introduced before and show their relationships.

Semantics $\equiv_{def}$ context information
Context Information $\equiv_{def}$ categories definition $\wedge$ class intension definition $\wedge$ geometry description
SFDS $\equiv_{def}$ syntax (FDS) $\wedge$ federated schema $\wedge$ context information
Semantic translator $\equiv_{def}$ Implementation of Proxy context

The above definition states that semantics is represented in SFDS as context information. Context information is the set of definitions in the context world view (i.e., categories definition, class intension definition, and geometry description). In order to resolve the heterogeneity among databases, SFDS is proposed. It consists of three layers, each layer resolves a type of heterogeneity. The syntactic heterogeneity is resolved by adopting the formal data structure. The schematic heterogeneity is resolved by introducing the concept of federated schemas. The semantic heterogeneity is resolved by an explicit association of context information with database schemas. When we design a semantic translator, we actually design a proxy context. The implementation of the proxy context, as an application program, results in the semantic translator.

4.7 Conclusions

Six types of abstraction were presented in this chapter. Categories definition, class intension definition, and geometric representation occur in the context world view. The other three types: geometric primitives, features, and class hierarchies occur at the computer representation. It was shown that differences in the definitions within the context world view, lead to three types of heterogeneity. These are semantic, schematic, and syntactic. The semantic heterogeneity is the main factor for schematic and syntactic heterogeneity.

The approach adopted in this research for resolving the heterogeneity has FDS as the syntactic layer and a combination of the federated database approach and the context mediation approach as the schematic and the semantic layers, respectively. In this respect the notion of proxy context was introduced, which is an abstraction of the common UoD of the contexts involved in information sharing. The proxy context is composed of context information and federated schema. The federated schema is not global to all GIS applications, instead, it is specific for an application domain. The semantic translator (which is a component of the SemWeb mentioned in chapter 3) is in fact an implementation of the proxy context.

In the next chapter a three-layer formal model of the proxy context is presented. The model is known as the semantic formal data structure, SFDS. Each layer is dedicated to resolve a specific type of heterogeneity.

Table 4-1 *A comparison between SFDS and other approaches to resolve heterogeneity.*

	Federated DB Systems	Multidatabase Systems	Context Mediation	SFDS
Explicit representation of and access to underlying data semantics. Semantics of data in underlying databases are explicitly represented and accessible by users.	No support. The semantics of the information providers is hidden from users. They only interact with the federated schema	No support. The export schema available to users might provide some hint to the semantics of underlying data. This, however, is inadequate, since semantic assumptions are frequently not obvious in the database schemes	Good support. This, conceivably, is possible by examining the mappings between individual database schema and the vocabulary of the context mediator	Good Support. Similar to context mediation. Furthermore it extends the notion of context mediation to include definitions of categories, intensions and geometry.
Explicit source selection subjected to users' preferences. Users retain the right to determining the databases which are to be queried to find the desired information.	No support. Access is restricted to pre-establishment (federated) schemes and ad hoc access to component databases is not permitted.	Good support. This is a feature of multidatabase systems. As a result, the export schema of each component database is generally available to users for search and to facilitate query formulation.	No support. Existing systems subscribe strongly to the need for a semantic model and do not normally provide explicit resource selection.	Good Support: The Resource Discovery Model (chapter 7) is complementary to SFDS which provides users with a mechanism for resource discovery and selection according to their preferences.
Explicit receiver heterogeneity. Representation of and access to user semantics. Receivers state their assumptions as to how data should be represented or interpreted instead of being	Limited support, via the definition of an export schemes, which only provides different ways of structuring the data.	Limited support. By having users define the conversions needed to comply with the assumptions.	Limited support. Context mediation can only differ in the vocabulary and general context information (coordinate system, scale, etc.).	Context information is tightly associated with the federated schema (chapter 5). Database designers establish the transformation once and for all, using functions which were predefined at the proxy

coerced into a unified world model.				context.
Automatic recognition and resolution (e.g., conversion) of semantic conflicts. The system recognizes semantic conflicts among component databases and users issuing the query, and performs the necessary resolution.	No support. Resolutions are established by the system administrator, a priori, in the underlying shared schemes.	No support. Users are left to their own devices in detecting semantic conflicts and defining the steps needed to convert from one representation to another.	Limited support. The focus on semantic heterogeneity has been limited and largely involves mapping from heterogeneous representations into a common interpretation in the context mediator	Limited support. The whole system architecture of the semantic translators provides database designers with capabilities to map between their schema and the translator. However, there is no automatic detection and resolution.
Conversion consideration in query optimization. Increase system throughput by considering how a query plan can be restructured to take into account the cost of converting from one semantic representation to another.	No support. Always requires component databases to map to a canonical representation prior to any further conversion.	Limited support. Users need to determine when conversions should take place by specifying these in the query. From the literature review there is no clear classification of methodologies for query optimization.	Limited support. Similar to multidatabase systems	Limited support. The terms used in the query processing are converted into global ones in the proxy context. These are then converted into the terms used by the information resource. There is no cost consideration for query optimization.

References

Ahmed, R., Smedt, P.D., Du, W., Kent, W., Ketabchi, M.A., Litwin, W.A., Raffi, A., and Shan, M.C., 1991. *"The Pegasus Heterogeneous Multidatabase System"*. IEEE Computer 24, 12 (Dec. 1991), pp. 19-27.

Arens, Y., and Knoblock, C.A., 1992. *"Planning and Reformulating Queries for Semantically Modeled Multidatabase Systems"*. Proceedings of the 1st International Conference on Information and Knowledge Management, 1992, pp. 92-101

ASTM (American Society for Testing and Materials) 1985. *"Annual Book of ASTM Standards"*. Section 4 construction, vol. 04.08, Soil and Rock; building stones. Philadelphia, USA., 972 pages, ISBN 0 8031 0654 8.

Bishr, Y., Molenaar, M., and Radwan, M., 1996. *"Spatial Heterogeneity of Federated GIS in a Client/Server Architecture"*. In Proceedings of the GIS'96, Vancouver, Canada.

Buehler, K., 1996. *"The Open Geodata Interoperability Specification Part II: Abstract Specification"*. Draft version. Open GIS Consortium, Inc., 35 Main Street, Suite 5, Wayland, MA 01778, USA.

Castellanos, M., and Saltor, F., 1991. *"Semantic Enrichment of Database Schemas: An Object Oriented Approach"*. First International Workshop on Interoperability in Multimedia Systems, April 7-9, 1991, Kyoto, Japan, pp. 71-78.

Collet, C., Huhns, M.N., and Shen, W.M., 1991. *"Resource Integration Using a Large Knowledge Base in Carnot"*. IEEE Computer, Vol. 24, No. 12, December 1991, pp. 55-62.

Engelen, V.W.P., van, and Ting-Tian, W., (eds.) 1995. *"Global and National Soils and Terrain Digital Databases - SOTER"*. Procedures Manual. Revised Edition, International Soil Reference and Information Centre (ISRIC), Wageningen, The Netherlands, 125 pages, ISBN 90 6672 059 X.

Fang, D., Hammer, J., and McLeod, D., 1991. *"The Identification and Resolution of Semantic Heterogeneity in Multidatabase Systems"*. Proceedings First International Workshop on Interoperability in Multidatabase Systems, 1991, U.S.A., pp. 52-59.

Kent, W., 1991. *"The Breakdown of the Information Model in Multi-Database Systems"*. Sigmond Record, December 1991, Vol. 20, No. 4, pp. 10-15

Landers, T., and Rosenberg, R., 1982. *"An Overview of Multibase"*. Proceedings of the 2nd International Symposium for Distributed Databases (1992), pp. 153-183.

Leeuwen, J. van, 1990. *"Formal Models and Semantics"*. Handbook of Theoretical Computer Science. The MIT Press, 1990 - XIV, 1237 pages. ISBN 0 444 88074 7 (Elsevier), ISBN 0 262 22039 3 (MIT Press).

Litwin, W., and Abdellatif, A., 1986. *"Multidatabase Interoperability"*. IEEE Computer, Vol. 19, No 12, December 1986.

March, T.S., (ed.), 1990. *"ACM Computing Surveys, Special Issue on Heterogeneous Databases"*. Vol. 22, No. 3, September 1990. ACM Press, 11 West 42 Street, New York, NY 10036. ISSN 0360-0300.

Molenaar, M., 1989. *"Single Valued Vector Maps - a Concept in GIS"*. Geo-Informationssysteme, Vol. 2, No. 1, 1989, pp. 18-26.

Molenaar, M., and Janssen, L.L.F., 1994. *"Terrain Objects, Their Dynamics and Their Monitoring by the Integration of GIS and Remote Sensing"*. (Ebner, H. et al., eds.), Spatial Information from Digital Photogrammetry and Computer Vision, Proceedings ISPRS Comm.III, Int. Archives of Photogrammetry and Remote Sensing , Vol. 30 part 3, München, Germany, pp. 585-591.

Molenaar, M., 1995. *"An Introduction into the Theory of Topologic and Hierarchical Object Modelling in Geo-Information Systems"*. Lecture Notes, Department of Land-Surveying & Remote Sensing, Wageningen Agricultural University, The Netherlands, 186 pages.

Molenaar, M., 1996. *"A Syntactic Approach for Handling the Semantics of Fuzzy Spatial Objects"*. In Geographic Objects with Indeterminate Boundaries. (Burrough, P.A., and Frank, A.U., eds.). Taylor and Francis, London, 1995, pp. 207-224.

Pilouk, M. , 1996. *"Integrated Modelling for 3D GIS"*. Ph.D. Thesis, Wageningen Agricultural University, 200 pages. ISBN 90 6164 122 5.

Sheth, A., and Larson, J., 1990. *"Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases"*, ACM Computing Surveys, Vol. 22, No. 3, September 1990, pp. 183-236.

Sims, R.A., Corns, I.G.W., and Klinka, K., (eds.) 1996. *"Global to Local, Ecological Land Classification"*. Kluwer Academic Publishers Group, P.O. Box 322, 3300 AH Dordrecht, The Netherlands, 372 pages, ISBN 0 7923 3966 5.

Urban, S.D., and Wu, J., 1991. *"Resolving Semantic Heterogeneity through the Explicit Representation of Data Models Semantics"*. SIGMOD Record, Vol. 20, No. 4, December 1991, pp. 55-58.

Ventrone, V., and Heiler, S., 1991. *"Semantic Heterogeneity as a Result of Domain Evolution"*. SIGMOD Record, Vol. 20, No. 4, December 1991, pp. 16-20.

CHAPTER

5 The Reference Model of SFDS

"Remember the waterfront shack with the sign FRESH FISH SOLD HERE. Of course it's fresh, we're on the ocean. Of course it's for sale, we're not giving it away. Of course it's here, otherwise the sign would be someplace else. The final sign: FISH."

Peggy Noonan (b. 1950),
What I Saw at the Revolution, ch. 4 (1990).

5.1 Introduction

In the previous chapter three types of heterogeneity were defined. These are syntactic, schematic, and semantic heterogeneity. The notion of proxy context was presented as a mediator between two or more contexts that share a common UoD. A semantic translator, shown in Figure 3-5, is an implementation of the proxy context. To resolve the syntactic, schematic, and semantic heterogeneity, the semantic formal data structure, SFDS, which is developed in this work, is proposed. SFDS consists of three layers, the syntactic layer, the schematic layer, and the semantic layer. The schematic and the semantic layers are considered the main contribution of this research.

Section 5.2 presents the general characteristics of SFDS. The syntactic layer is described in section 5.3. In this section a brief outline of the formal data structure, FDS, is presented. FDS formally describes the syntax of geometric and thematic aspects of spatial objects. The schematic layer is presented in section 5.4, where a reference model is explained. The reference model will be used to describe federated schemas of the underlying semantic translator. The semantic layer is the third layer in SFDS and is described in section 5.5.

The semantic layer shows different types of the context information and the methods to associate them with database schemas. The chapter is concluded in section 5.6.

5.2 Characteristics of SFDS

The semantic formal data structure, SFDS, has three layers: the syntactic layer, which takes the formal data structure as its foundation; the schematic layer where we propose a reference model, which any federated schema can be attached to; and the semantic layer, which includes the context information. The general characteristics of SFDS are:

- At the first layer, the syntactic heterogeneity is resolved. SFDS adopts the formal data structure, FDS.
- At the second layer, the schematic heterogeneity is resolved. SFDS adopts the concept of federated schemas, to which each database schema should map. The design of a federated schema is specific for a particular application domain.
- The context information is needed to map between the heterogeneous database schemas. For this reason, the semantic layer provides a mechanism to associate this information with the federated schema.

5.3 The Syntactic Layer of SFDS

The formal data structure is the syntactic layer of SFDS which is fundamental to all representations of the geodata in GIS. This research requires that all component databases, as well as the semantic translator, comply with the syntax of FDS. In FDS an object that belongs to a class has an identifier, for a unique identification in the database, as well as geometric and thematic descriptors, for which see Figure 5-1. A class has a label and a list of attributes, which characterizes that class. A database object is a member of some class and has the

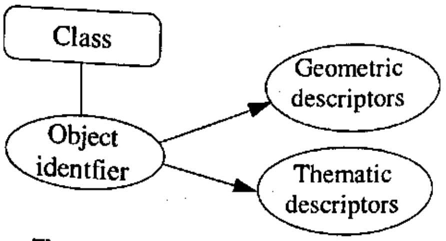


Figure 5-1 Representation of spatial objects in FDS.

attribute structure of the class it belongs to. For example, a house can be defined by a unique identifier in the database (e.g., a house number and address), has a geometric description (e.g., bounding rectangle), and has thematic descriptors (e.g., number of rooms and owner).

FDS defines a syntax for both the geometric and the thematic descriptors of spatial objects. A brief discussion of the geometric and the thematic formalism is provided in sections 5.3.1, 5.3.2, respectively. The formalism of the geometric syntax is given in a 2-D planar graph. Only a summary of the syntax of the vector maps is presented here. More details can be found in [Molenaar, 1989; Molenaar, 1991; Molenaar, 1994; Molenaar, 1996 a, b, c, d].

5.3.1 The Geometric Syntax of FDS

The formal data structure formalizes the syntax for spatial objects in a vector database. In FDS the same syntax can be applied to raster structure and other tessellations by considering them as faces in a planar graph. The spatial structure is expressed in terms of nodes, edges, and faces. The general characteristics are listed, and shown in Figure 5-2:

- The model has three geometric primitives, which are nodes, edges, and faces.
- Three types of complex objects: point objects which are defined by nodes, lines which are collections of edges, and area objects which are geometrically described by a collections of faces.
- An edge has a begin and an end node.
- An edge has one left and one right face.
- Two edges join at no more than one node.
- Two line objects can cross, or intersect, each other at a node, one line object is upper and the other is lower.

Position information is given by the coordinates of the nodes and the geometry is described by means of the geometric primitives. Points and lines are dealt with as nodes and edges, respectively. Therefore, the underlying mathematics for the geometric description of spatial objects in a vector map is provided by graph theory.

The geometry of a vector database is represented in a planar graph $G(N,E)$ where

- $N = \{n_1, \ldots, n_l\}$ is the set of nodes in a context
- $E = \{e_1, \ldots, e_m\}$ is the set of edges in a context, where

  $\forall e_j = \{e_j = (n_p, n_q)\}$ is an ordered pair where n_p, $n_q \in N$, and
- We say that $\mathcal{F}$ is the set of faces generated by

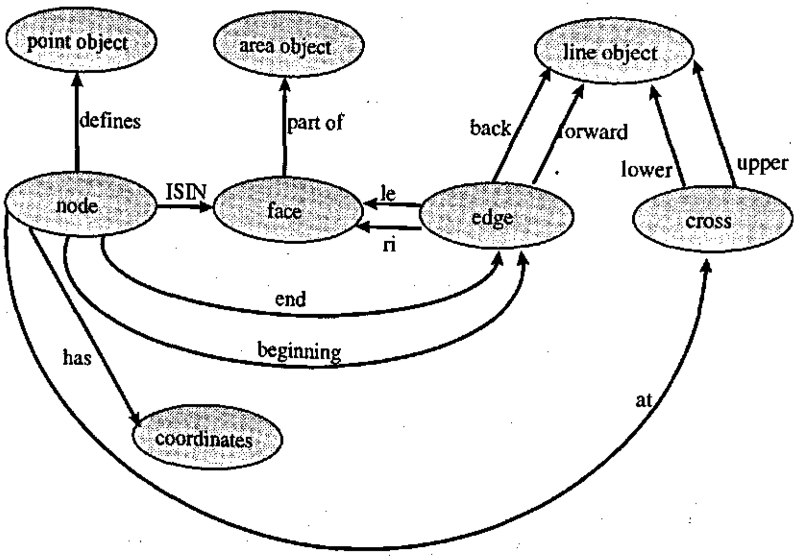


Figure 5-2 Formal data structure.

  (N, E) in a planar graph.

FDS identifies six types of topologic relationships between pairs of elementary objects (only a list is provided):

- Point object - point object
- Line object - point object
- Area object - point object
- Line object - line object
- Area object - line object
- Area object - area object

5.3.2 The Thematic Syntax of FDS

Objects in FDS have thematic and geometric descriptors. The geometric descriptors are briefly shown in the previous section. Objects in a context are distinguished on the basis of their different characteristics. In most GIS applications these differences are thematic. Two objects are distinct, if their thematic descriptors are not equal. Objects with similar characteristics are collected in classes. Criteria are formulated for each class to specify when an object is a member of that class. This is known as intension definition, as defined in section 4.3.2. The following formalism assumes that classes and objects are defined within a context.

Let C be a class which has a set of attributes A.
We use the national convention
$LIST\ (C) = \{A_1,\ \ldots,\ A_r,\ \ldots,\ A_n\}$ *in Cont*

If an object O passes a test formulated in a decision function for a class C, it will be a member of that class and that will be expressed by the membership function:
$M(O,\ C)$ = *True* if O is a member of C

= *False* otherwise in *Cont*

The attribute structure of objects is determined by the class to which they belong, so that each object has a list containing one value for every attribute of its class. An object then takes the attribute structure of its class.

$M(O,\ C)$ = True implies that
$VLIST(O) = \{a_1,\ \ldots,\ a_r, \ldots,\ a_n\}$ in *Cont*
where $a_r = A_n(O)$ is the value of A_n for object O
and $A_r \in LIST(C)$
and $a_r \in DOMAIN\ (A_r)$

The extension of a class is the set of all the objects that belong to it, hence
$Ext(C) = \{O\ |\ M[O,\ C] = True\}$ in *Cont*

Classes C_i and C_j are semantically distinct, if they have different attribute structures, i.e.,
$LIST(C_i) \neq LIST(C_j)$ in *Cont* where $i \neq j$

Classes within a context are exhaustive, which means that all objects identified in a context must belong to some class. Classes which are semantically distinct within a context, are disjunct. The notion of disjunction implies that an object can belong to only one class within a context. The two conditions can then be defined respectively as follow:

- $(\forall O \mid O \in Cont)\ (\exists C \mid C \in Cont) \Rightarrow M[O, C] = True$
- Let P be the set of all classes in $Cont$, i.e., $P = \{C_1, \ldots, C_c\}$ and
- $(C_j, C_k \in P \mid C_j \neq C_k) \Rightarrow Ext(C_j) \cap Ext(C_k) = \emptyset$

A superclass has the list of common attributes between a set of classes. The classes with the common attributes are known as the subclasses of the superclass. This implies that a subclass has the attributes of its superclass (i.e., inheritance) in addition to its own attributes. Hence, the attribute value list of an object contains the values of the attributes of its class and the superclass. The extension of a subclass is also part of the extension of its superclass but not vice versa. For example, a superclass *Transportation* can have *Roads* and *Railways* subclasses. The extension of roads is also a part of the extension of transportation but the extension of transportation is not necessarily the extension of roads. The above discussion can be formalized as follows:

Let C_s be a superclass of C_k, then
$O \in Ext\ (C_k) \Rightarrow O \in Ext\ (C_s)$
If $O \in$ Ext C_k then
$VLIST(O) = \{a_1, \ldots, a_r, \ldots, a_n\}$ in $Cont$
$A_r \in LIST(C_k)$
$a_r = A_r(O)$ is the value of A_r for object O
$A_r \in \text{List}(C_k) \cup \text{List}(C_s)$
$\text{Ext}(C_k) \subseteq \text{Ext}(C_s)$

The relationship between the geometric description mentioned in section 5.3.1 and the thematic description is formalized in FDS. Only the relationship between faces and the thematic description is shown here, the same can be applied to point and line objects. Based on the requirement that each face in a context is related to exactly one area object, each edge is related to at most one line object and each node to at most one point object. Therefore, the objects form the geometric partition of the underlying UoD. The relationship between the geometric description and classes of the objects can then be found through the extensions.

Let the set of faces related to a class C_p in a context be:
$\mathcal{F}_{cp} = \bigcup_{oi \in cp} \mathcal{F}_{oi}$ in *Cont* where $\mathcal{F}_{oi}$ is the set of faces of object O_i
Let the set of faces related to a class C_q in the same context be:
$\mathcal{F}_{cq} = \bigcup_{oj \in cq} \mathcal{F}_{oj}$ in *Cont* where $\mathcal{F}_{oj}$ is the set of faces of object O_j

·The fact that two area objects are disjunct, i.e.,
$\mathcal{F}_{oi} \cap \mathcal{F}_{oj} \neq \emptyset$, and
the extensions of two semantically distinct classes are also disjunct, i.e.,
$\mathrm{Ext}(C_p) \cap \mathrm{Ext}(C_q) = \emptyset$
implies that the faces of the two classes are disjunct, i.e.,
$\mathcal{F}_{cp} \cap \mathcal{F}_{cq} = \emptyset$

After defining the relationship between thematic and geometric characteristic of spatial objects, the model presents the concepts of generalization and aggregation hierarchies [Smith et al., 1977]. There are four strategies for generalization and aggregation:

1) *Geometry-driven generalization*: The execution of the strategy is mainly dependent on the geometric description of the spatial data. Such a type of generalization is mainly applied in cartographic generalization.

2) *Class-driven generalization and aggregation*: In this strategy the aggregation is executed on the basis of the thematic information of the spatial objects, while the generalization is based on relationship between class intension. Adjacency relationship is mostly required in aggregation hierarchies. Adjacent objects which are of the same type are generalized to more general classes. For example adjacent area objects of the classes forest and grass land are aggregated to form larger objects of natural vegetation.

3) *Function-driven aggregation*: Objects from different classes at a low aggregation level, i.e., elementary objects, defined in one context, are aggregated to form a new complex object of another class. Elementary

objects have part-of relationship with complex objects from higher aggregation levels. For example homogeneous geologic structures and soil types are aggregated to form soil mapping units.

4) *Structural generalization*: In this generalization strategy the aim is to simplify the description of a spatial system, while leaving the overall structure intact. For example in a utility database, the water pipes network can be generalized into main water pipes by eliminating house connections and maintaining the main pipes.

5.4 The Schematic Layer of SFDS

The second layer of SFDS is the schematic layer. In this layer, the federated schema is defined. The federated schema is dependent on the underlying contexts that will share their data. It is proposed that the federated schema should be designed in such a way that it provides information sharing of a certain application. For example, if the purpose of a set of contexts is to share road and hydrology information, we design two federated schemas, and hence two semantic translators, one for hydrology and the other for road information. Figure 5-3 shows the reference model of SFDS, which is used to describe the underlying federated schema. In the sequel, the reference model and its integrity constraints are described, this is supported by an example of a federated schema attached to the reference model.

5.4.1 The Reference Model

The reference model of SFDS consists of a proxy context *Pcontext*, proxy hierarchy *Phierarchy*, proxy class *Pclass*, and a proxy attribute *Pattribute*, as illustrated in Figure 5-3. The relationship between these elements is an association relationship (or *member-of* relationship). A *Phierarchy* is a member-of only one *Pcontext*. A *Pcontext* can have more than one *Phierarchy*. *Pclass* is a member-of *Phierarchy*. A *Pclass* cannot be a member-of more than one *Phierarchy*, while a *Phierarchy* can have more than one *Pclass*. Similarly, *Pattribute*, which is a member of *Pclass*, belongs to only one *Pclass*.

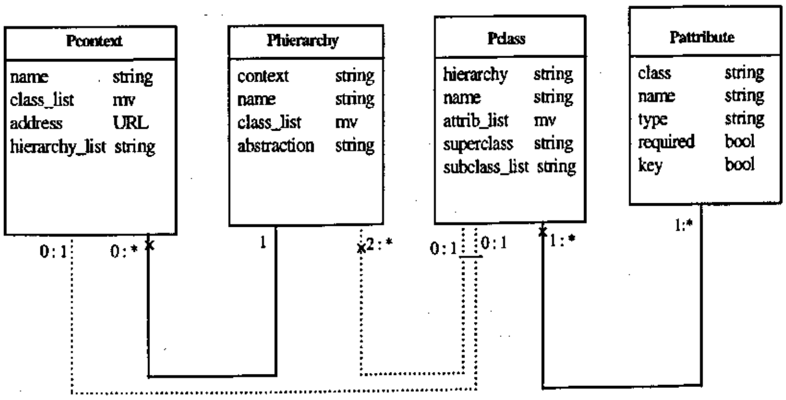


Figure 5-3 The reference model of the semantic translator.

The notation shown in Figure 5-3 is based on the object-oriented system analysis model, OSA. The concepts of OSA are based on formal definitions of system data and behavior modeling [Embley et al., 1992]. The numbers shown near the connection of the objects are the cardinality constraints. These are non-negative integer numbers in the form *min:max*. The star designates an arbitrary non-negative number. The dotted line, shown in Figure 5-3, presents a special type of association. It represents a circumstantial association, as opposed to an essential association, which is represented as a continuous-solid line. A circumstantial association describes a relationship which depends on other conditions in the reference model. These conditions are not always satisfied. For example, *Pclass* is a member-of *Pcontext*, if it does not occur in any *Phierarchy*.

Let S be the federated schema which is described in the reference model.

The domain with respect to *S* is *Bool, string, mv, URL*
mv is a set of *strings*, and
URL= <prot>://<server>/<pathname>

Where *prot* is an Internet protocol : FTP, HTTP, SMTP, etc.
server is the IP address of the Internet server.
pathname is the location of the file name on the server

f: *D* → *R* is a bijective function with respect to *S*, where
D = *S* ∪ M ∪ *U* ∪ B

S = {name, context, abstraction, hierarchy, superclass, subclass, class}
M = (attrib_list, class_list, hierarchy_list)
U = {address}
B= {type, key}

$R = R_1 \cup R_2 \cup R_3 \cup R_4$
$R = string \cup mv \cup URL \cup Bool$
$\forall s \in S : f(s) \in R_1$
$\forall m \in M : f(m) \in R_2$
$\forall u \in U : f(u) \in R_3$
$\forall b \in B : f(b) \in R_4$

The next constraints pertain to the relationship between the schema elements of the proxy context. We can establish the constraints that will be applied on hierarchies, classes and attributes of the federated schema.

The notation X.Y indicates a schema element X which has the attribute Y.

- $\forall$Pcontext.name (hierarchy-list | hierarchy-list = {Phierarchy.name})
- $\forall$Phierarchy.name (class-list | class-list $\subseteq$ {class.name})
- $\forall$Pcontext.name (class-list | class-list = {class.name})

The above states that the set of Pcontext.hierarchy-list must contain all the elements of the set of *Phierarchy.name* at any state of the system. The list of classes in a hierarchy must be a proper set of the class names in the underlying proxy context. The class-list must contain all the class names in the underlying Pcontext.

Similarly we can state the constraints that apply to the attributes and their relationship with classes.

- $\forall$Pclass.name (attrib_list | attrib_list $\subseteq$ {Pattribute.name})
- $\forall$Pattribute.name (Pclass | class $\in$ {Pclass.name} $\wedge$ name $\in$ Pclass.attrib_list)

- ∀Pclass.name (hierarchy | hierarchy ∈ Pcontext.hierarchy_list).
- ∀Pclass.name (subclass | subclass ∈ Pcontext.class_list).
- ∀Pclass.name (superclass | superclass ∈ Pcontext.class_list).

Example

Figure 5-4 shows the federated schema designed for sharing road information. Although the federated schema is not complete, it suggests that federated schemas are not necessarily complex as mentioned in section 4.6. Figure 5-5, shows the federated schema after it has been described in the reference model. The class *pavement* has two subclasses, *street* and *motorway.* The class *pavement* has a list of attributes (only the attribute *asphalt* and *speedlimit* are shown). The three classes, *pavements, street,* and *motorway* form the hierarchy *road.* The *road* hierarchy belongs to the proxy context *transportation.* The federated schema embedded in the reference model, forms the thematic description of the proxy context. It is consistent with the syntax of the thematic description in FDS, mentioned in section 5.3.2.

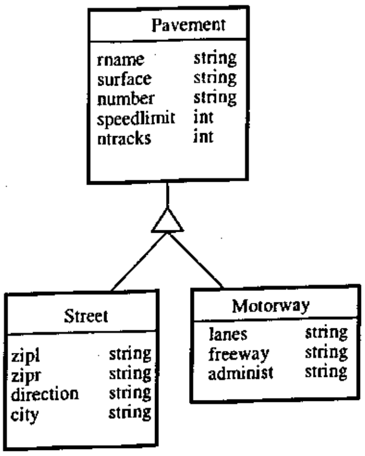


Figure 5-4 The federated schema for sharing road network data.

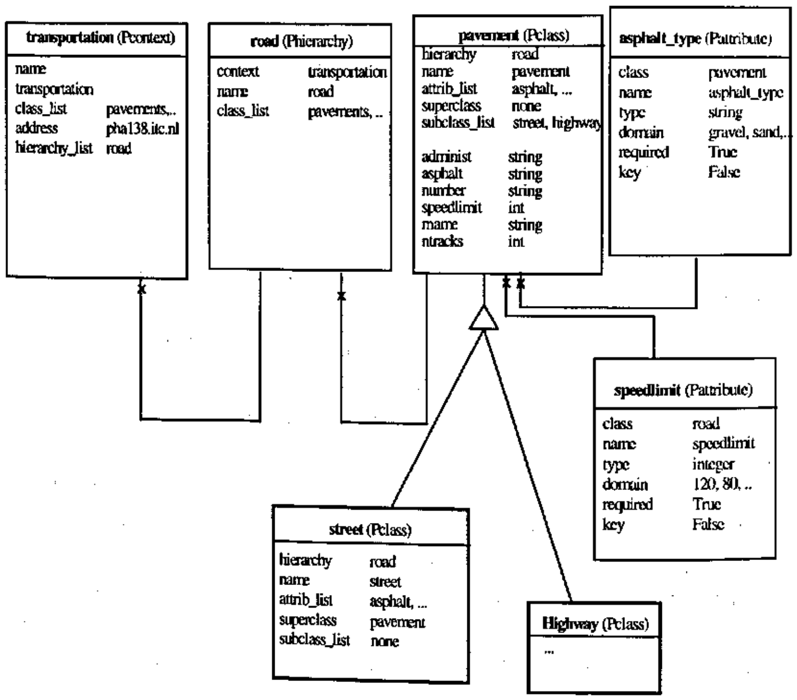


Figure 5-5 The federated schema described in the reference model.

5.4.2 Manipulating the Reference Model

After introducing the reference model of the proxy context, a set of functions for handling the model is presented here. The functions are defined at an abstract level and they assist the database designer to manage and manipulate the proxy context while designing the semantic translator. Invoking the functions follows the usual message passing convention of the object-oriented programming where a message expression is sent to an object to trigger a method which returns a result. The functions are expressed with the use of the following syntax.

FunctionName(schemaElement, {Parameters}) → Return Value
where the schema element is either Proxy context, proxy hierarchy, proxy class, or proxy attribute.

The Functions are classified as:

- Functions on proxy context

- Functions on proxy hierarchies
- Functions on proxy classes
- Functions on proxy single-valued attributes
- Functions on proxy multi-valued attributes
- Functions on proxy instances
- General functions which operate on any of the above.

General functions

- GetName(X)
  Where X ={Pcontext, Phierarchy, Pclass, Pattribute}
  Returns the name of the schema element X

- GetCLassList(X)
  Where X ={Pcontext.name, Phierarchy.name}
  The function returns the list of classes in the proxy context (the list of classes in the federated schema) or in the specified proxy hierarchy.

- GetOntology(X)
  Where X ={Pcontext.name, Phierarchy.name, Pclass.name, Pattribute.name}
  Returns a list of ordered pairs of ontology definitions (ontology, value) associated with X (The issue of ontology will be discussed in chapter 6).

Functions on proxy context

- GetOwner (Pcontext.name)
  Returns the address of the owner of the underlying context. Owner is a unique name of the data provider.

Functions on proxy hierarchies

- GetAbstType(Phierarchy.name)
  Where name is the name of the Phierarchy
  Returns the class hierarchy type (association, generalization or aggregation).

Functions on classes

- GetAttrib(Pclass.name)

Returns the list of attributes of the class.

- GetHierarchy(Pclass.name)
  Returns the name of the hierarchy to which the proxy class belongs.

- GetSuper(Pclass.name)
  Returns the superclass of the proxy class. If the class has no supper class the value of name is null.

- GetSub(Pclass.name)
  Returns the subclass(es) of the proxy class. If the class has no super class the value of name is null.

Functions on single-valued attribute

- GetValue(Pattribute.name)
  Returns the value of the proxy attribute.

- GetType(Pattribute.name)
  Returns the type of the proxy attribute. This function also applies on multi-valued attributes.

Functions on multi-valued attributes

- GetNum(Pattribute.name)
  Return the number of elements of the multi-valued attribute

- GetType(Pattribute.name)
  Returns the type of the proxy attribute.

- GetMultVal(Pattribute.name, num)
  Returns a value from a multi-value attribute which has the number specified by num.

5.5 The Semantic Layer in SFDS

The key to enabling interoperability at the semantic layer is the explicit representation of context information. Context information is derived from the discipline world view mentioned in section 4.2, and shown in Figure 4-1 (categories definition, objects definition, and geometric definition).

Figure 5-6, shows an extension to the classic representation of geographic objects. In the proposed model, objects have geometric and thematic descriptors. Context information is attached to the classes, i.e., context information is defined at the intensional level. Context information forms the semantics of that class.

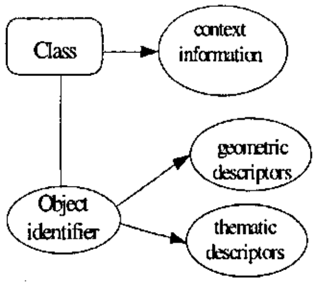


Figure 5-6 Semantics are defined at the intensional level.

The context information differs from the other descriptors as they are defined at the class level, and hence all instances of that class will have the same context information. In this case, an object, retrieved from a data provider, can semantically belong to a class in the proxy context, although it has a different geometry and/or part of its thematic descriptors are different. The types of semantic heterogeneity are illustrated in section 5.5.2 together with some examples. A proxy context should satisfy the following conditions:

1) Semantics are only defined at the intensional level (not the extensional level).

2) Proxy classes are media by which objects are transferred from one database to the other.

3) A proxy element (hierarchy, class, or attribute) must be semantically similar to at least one element at the schema of the data provider and one element at the schema of the receiver.

4) An object which belongs to a class, is a member of a proxy class if both the class and the proxy class are semantically similar. Consequently, the membership of an object to a proxy class is determined on the basis of the similarity in context information, i.e., semantics, not only on its geometric representation or attributes.

Statement 4 is an extended interpretation of the FDS requirement, which states that "*objects inherit their attribute structure from their corresponding classes*", [Molenaar, 1996]. An important characteristic of the proxy context is that objects, which are members of a class in a context, retrieved to a proxy context can be attached to a proxy class and inherit the attribute list of that proxy class. This can be illustrated as follows:

We will use the operator $\cong$ to indicate semantic similarity

Let Cont be a context.
$\{C_1, \ldots, C_m\}$ is the set of all classes in Cont, and
$LIST(C) = \{A_1, \ldots, A_k\}$ is the set of attributes of a class C
1

If an object O passes a test formulated in a decision function for a class C_i then it will be a member of that class and that will be expressed by the membership function:
$M(O, C)$ = True if $O \in C$
= False otherwise
then $VLIST(O) = \{a_1, \ldots, a_k\}$ in Cont. **2**

$\{PC_1, \ldots, PC_n\}$ is the set of all classes in PCont, and
$LIST(PC) = \{PA_1, \ldots, PA_l\}$ is the set of attributes of a class PC
3

The relationship between PCont and Cont is defined as
$PCont \subseteq Cont \Rightarrow (\exists C \in Cont)$
$(\exists PC \in Pcont) \Rightarrow C \equiv PC$, then
$Ext\ (PC) \subseteq Ext\ (C)$
4

From **2** and **4**
$O \in C \Rightarrow O \in PC$
5

However according to **2** the attribute value list of O is
$VLIST(O) = \{a_1, \ldots, a_k\}$ in Cont

Hence we define a function *Map* which map object O from class C in Cont into Class PC in Pcont.
$Map\ (O\{a_1, \ldots, a_k\}) \Rightarrow (O\ \{pa_1, \ldots, pa_l\})$
Which means that after the mapping
$VLIST(O) = \{pa_1, \ldots, pa_l\}$ in Pcont

Once the object O is mapped to Pcont, then it should satisfy the syntax of FDS as mentioned before.

An object is mapped from a context to a proxy context based on the similarity among their classes. The *Map* function can be considered as a membership function which transforms the membership of an object from a class in Cont into another semantically similar class in Pcont, or vice versa. In other words we can say that:

```
If C ≅ PC then
M(O, C) = M(O, PC)
```

The formalism only introduced the case where two classes are semantically similar and there is one-to-one mapping between them. However, there are other different cases, for example, a class in Cont might be semantically similar to an attribute in Pcont. In this case the *Map* function will map the class into an attribute. In chapter 6 the different cases of semantic similarity which may occur among schema elements are introduced in more detail.

5.5.1 Representing Context Information

Resolving the heterogeneity among schemas requires knowledge about what each schema element means. The proposal made by Doyle and Kerschberg to encapsulate data and knowledge is used here [Doyle et al., 1991]. Schema elements and context information can be encapsulated into an abstract object type to make it possible to map between database schemas. In SFDS data/knowledge packets are formed, with the use of triples of the form <schema_element, context information, operation>, where *schema_element* is a class or an attribute, *context information* is represented as first order predicates, and *Operations* are rules of inference which take their values from context information. To provide interoperability at the semantic level, the following conditions must be satisfied:

1) All sources and receivers which share their information must describe their contexts explicitly.
2) All queries and retrieved information must be routed through the semantic translator.

The method applied makes use of two types of predicates. The first type of logic predicates, which is used to represent context information, is the *descriptive* predicate. A descriptive predicate is actually associated with a particular schema element. The second type is the assertion predicates. The following example illustrates the concept.

Example

Consider that contexts $Cont_1$ and $Cont_2$ have soil information. A proxy context PCont is designed to share soil information between $Cont_1$ and $Cont_2$. $Cont_1$ has land-use suitability information which is required by $Cont_2$, as shown in Figure 5-7. It is then required to map the required information from $Cont_1$ to PCont and then to $Cont_2$.

$Cont_1$: Class *soil-unit* has attribute *suitability* where its domain Dom_1 = {1,2,3}, where 3 indicates the highest suitability
PCont: Class *soil-unit* has attribute *suitability* where its domain *PDom* = {1,2,3}, where 1 indicates the highest suitability.

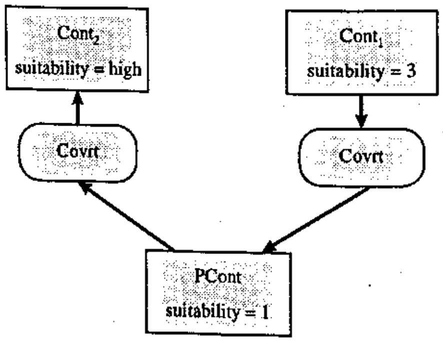


Figure 5-7 Mapping between domains in two contexts.

Suitability information cannot be exchanged between the two databases, unless more information is provided about their semantics. Schema_element, context information, and operation triples can be presented as follows:

```
In Cont1
Schema_element = soil-unit.suitability
Context information = ∀soil-unit(suitability ∈ Dom1
∧ Dom1 = {1,2,3}∧ 3 > 2 > 1)

In PCont
Schema_element = soil-unit.suitability
Context information = ∀soil-unit(suitability ∈ PDom
∧ PDom = {1,2,3}}∧ 1 > 2 > 3)
```

An operation can be defined to map between the two contexts as follows:

```
Operation = If suitability ∈ Dom1 then
Map(suitability, Cont1, PCont) ⇒ PDom
```

Where Map is a function which maps the elements of Dom_1 in $Cont_1$ into PDom in PCont. Similarly a mapping can be established between PCont and $Cont_2$.

Let Cont represent context information and, $cont_i$ be one of its aspects, and
let V_i be the value of $cont_i$, so that $1 \leq i \leq K$,

A schema element is defined by a set of ordered pairs
Cont = $<(cont_1, V_1), \ldots(cont_k, V_k)>$ which characterize the semantics of an application domain.

Which is the general syntax to represent context information.

Let Γ represent a schema element. We write $\Gamma((cont_1, V_1) \wedge \ldots \wedge (cont_k, V_k))$ to associate context information and their values with Γ.

The above syntax shows the way the context information is represented. The above example has the same syntax. Gamma, Γ, corresponds to the schema-element. The functions which convert schema elements from a context to another schema element in the proxy context can be outlined as follows:

Let *Covrt* be a function which maps Γ from some context Cont, to $\Gamma`$ in another context Cont`, then
covrt (Γ) $(((cont_1, V_1) \wedge \ldots \wedge (cont_k, V_k)) \wedge ((cont`_1, V`_1) \wedge \ldots \wedge (cont`_k, V`_k))) \rightarrow \Gamma`$
where *Covrt* takes its arguments from Cont and Cont`.

Note that the underlying schema elements in the two contexts must be semantically similar.

5.5.2 Types of Context Information

Semantic heterogeneity can occur due to differences in the categories definition, differences in object definition, and differences in the geometric description. These types of semantic heterogeneity mainly occur due to the fact that objects play different roles in different databases. In this section, a discussion and some examples of the types of semantic heterogeneity are shown. This will help to understand the types of context information, as defined in this research, which are needed to map between contexts and a proxy context, and vice versa. Database designers use context information to map between their contexts and the proxy context. The mapping between two schemas is a schematic problem, however, the knowledge needed to provide a correct mapping is semantic.

5.5.2.1 *Context Information for Categories Definition*

Different classification criteria can be defined to classify the same real world features. For instance, roads can be classified on the basis of the pavement type, number of lanes, or the type of administrative organization (federal government or state). Alternatively, the classification can also be based on the road type (highway, secondary road, etc.). Context information should, then, indicate the criteria for classification.

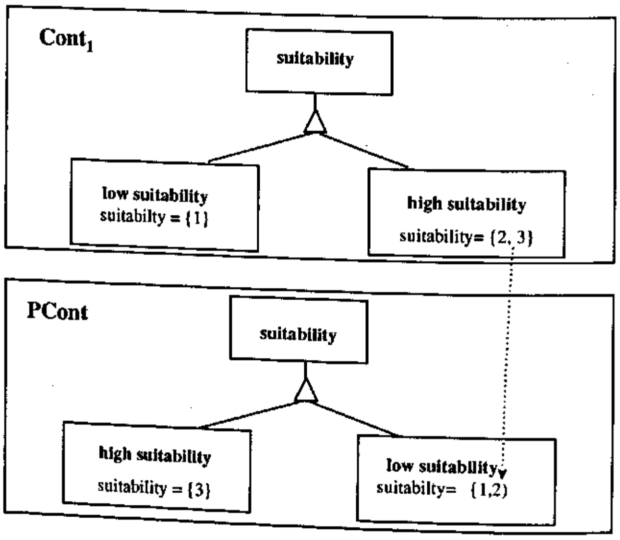


Figure 5-8 An object with suitability = 2 and member of class high suitability in Cont$_1$ is associated with class low suitability in Pcont.

5.5.2.2 Context Information for Class Intension Definition

Context information for class intension definition provides rules to associate objects with classes. Figure 5-8 shows a case of context information for object definition. Differences in the attribute domains between a context and a proxy context may cause objects to be members of classes with different semantics. An object in Cont₁ which is a member of the class high suitability and has suitability = 2 is mapped to the class low suitability in PCont.

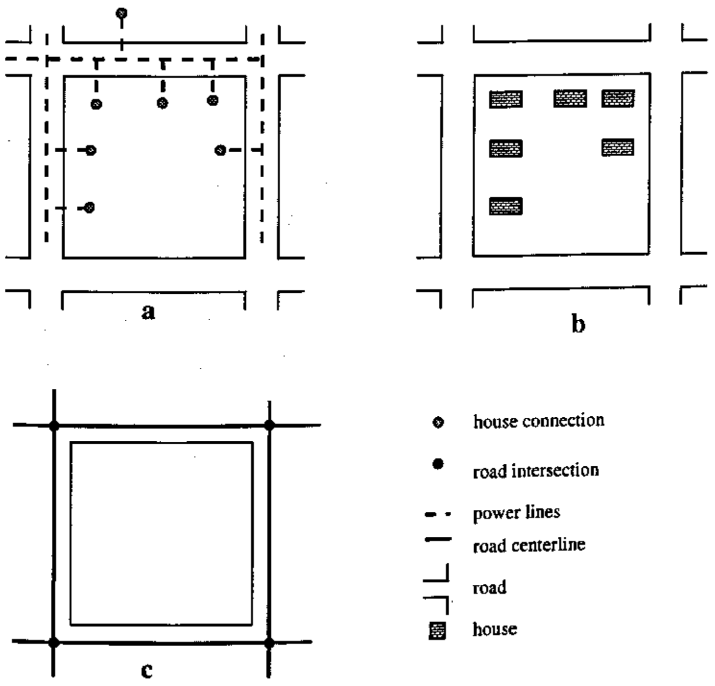


Figure 5-9 Difference in geometric representation due to semantic differences; a) utility context, b) cadastre context, c) road management context.

5.5.2.3 Context Information for Geometric Description

Objects in different contexts, can have different geometric representations not only due to spatial resolution (i.e., scale) differences but also due to their role in their underlying context. For example, as shown in Figure 5-9, houses are area features in a cadastre database, point objects in a utility database, and aggregated to form a block in a database for road management. Roads are defined by their boundaries in the utility and the cadastre databases, while they are represented by their centerline in the road management database.

The topologic relationships which are required to associate between spatial objects (for the purpose of spatial analysis) are important factors in determining their geometric representation. In the same figure, in the context of road management, roads are represented by their centerline, because road intersections are needed for analysis. On the other hand, the utility context requires the road boundary in order to locate the underground utility accurately. Context information has to provide an explicit representation of the topologic relationships of objects as shown in the next section.

5.5.3 Geometry and SFDS

SFDS separates between schematic and geometric heterogeneity. In the previous sections we showed a mechanism that allows context information to be associated with schema elements to support the process of resolving schematic heterogeneity. From the geometric representation point of view, objects retrieved from a remote database can be subject to a geometric generalization process. For instance, houses which are represented as area features in the cadastre context will be generalized to point features when retrieved to the utility context.

```
In the proxy context no cartographic generalization
process is applied. This task is left to the data
receiver.
```

The above statement was introduced to minimize the data loss when mapping from an information resource to a proxy context and then to the user.

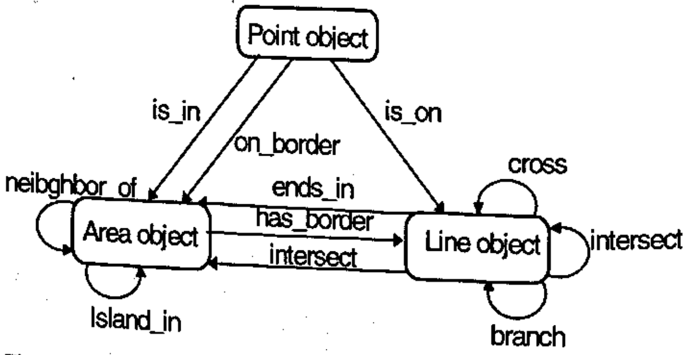


Figure 5-10 Topologic relationships among elementary objects (after Molenaar, 1991).

Generalization in GIS is a transformation process with the following two objectives [Peng, 1997]:

- To derive a new (digital) database with different (coarser) spatial, thematic, and/or temporal resolutions from existing databases, for a particular application.
- To enhance graphic representations of a database or part thereof, when the output scale cannot accommodate the data set of interest, for visualization purposes.

The issue of database generalization is implicitly tackled, when the semantic heterogeneity is resolved. Enhancing the graphic representation is not the intention of this work. Peng proposes a concept for automated generalization [Peng, 1997]. Automated generalization requires knowledge about the topologic relationships among the objects involved. Therefore, when an object moves to another database via the proxy context it is necessary to associate explicitly its topologic relationship with other objects in the original database. This approach allows automated generalization to employ the topologic relationships among the spatial objects deductively. Furthermore, another advantage of this approach is that when an object is retrieved, the system can automatically request the other topologically related objects in order to perform generalization or any other spatial process. Figure 5-10, shows the set of topologic relationships among the three types of objects (point, line, and area). Following this, topology can be associated with objects as follows:

```
island_in (OID1, OID2)
Where OID1 and OID2 are object identifiers of area
objects.
```

Similarly other topologic relationships, shown in the figure, can be presented. Object identifiers play an important role in such an approach. Object identifiers must be unique and persistent within all contexts. Object identifiers usually serve two purposes. Firstly, they provide unique reference to objects in the component databases, for instance for future update. Secondly, they are used to derive references to objects in the client/server environment. The issue of universal object identifiers is an active area of research in the field of transaction management in federated databases and client/server architecture. More information on this subject can be found in [Heiler et al., 1989; Khoshafian et al., 1986; Manola, 1993; Common Object

Request Broker, 1995]. In general, several prerequisites have to be considered during the development of universal object identifiers:

- The ability to provide a flexible mechanism to meet the requirements for accommodating both local and retrieved objects.
- The ability to guarantee correctness, that is, uniqueness and immutability.
- The ability to provide adequate performance without any consistency violations.

5.6 Conclusions

SFDS is presented in this chapter. It consists of three layers: syntactic, schematic, and semantic. The syntactic layer complies with the FDS. At this layer the formalism of the geometric and the thematic aspects of spatial objects are summarized. The other two layers are the main contribution of this research. The second layer is the reference model which is used to describe the federated schema of the application domain. The third layer is the semantic layer where context information is defined. The relationships between each layer are also presented.

The objective of this chapter was to introduce a concept for associating context information with schema elements. Examples of context information which are associated with schema elements are introduced. The context information allows the mapping between semantically related schema elements in the underlying context and the proxy context. The problem of finding semantically related schema elements will be the subject of the next chapter.

Although the issue of handling different geometric representations in heterogeneous databases is not the focus of this research, only a proposal to handle this situation is presented. Further investigation and elaboration is still required.

References

Common Object Request Broker (CORBA) - Architecture and Specification. Revision 2.0, July 1995, updated July 1996.

Doyle, W., and Kerschberg, L., 1991. *"Data/Knowledge Packets as a Means of Supporting Semantic Heterogeneity in Multidatabase Systems"*. SIGMOD Record, Vol. 20 No. 4, 1991, pp. 69-73.

Embley, D.W., Kurtz, B.D., and Woodfield, S.N., 1992. *"Object Oriented Systems Analysis - A Model Driven Approach"*. Yourdon Press Computing Series. Prentice-Hall, Inc., A Simon & Schuster Company, Englewood Cliffs, New Jersey 07632. 302 pages. ISBN 0 13 629973 3.

Heiler, S., and Blaustein, B., 1989. *"Generating and Manipulating Identifiers for Heterogeneous, Distributed Objects"*. (Rosenberg, J. and Kock, D., eds.). Persistent Object Systems, Springer-Verlag, London, 1989.

Khoshafian, S., and Copeland, G., 1986. *"Object Identity"*. In N. Meyrowitz, ed. OOPSLA '86 Conference Proceedings, ACM, Sept. 1986, published as SIG-PLAN Notices, 21:11, November 1986.

Manola, F., 1993. *"MetaObject Protocol Concepts for a "RISC" Object Model"*. Technical Report TR-0244-12-93-165, GTE Laboratories Incorporated, 1993.

Molenaar, M. , 1996 (a). *"A Syntactic Approach for Handling the Semantics of Fuzzy Spatial Objects"*. In: Geographic Objects with Indeterminate Boundaries, P.A. Burroug and A.U. Frank (eds.), Taylor and Francis, London pp. 207-224.

Molenaar, M. , 1996 (b) .*"The Role of Topologic and Hierarchical Spatial Object Models in Database Generalization"*. In: Methods for the generalization of geo-database, M. Molenaar (ed.), Netherlands Geodetic Commission, New Series, Nr. 43, Delft, pp. 13-36.

Molenaar, M., 1989. *"Single Valued Vector Maps - a Concept in GIS"*. Geo-Informationssysteme, vol. 2, no. 1, 1989, pp. 18-26.

Molenaar, M., 1991. *"Status and Problems of Geographical Information Systems. The Necessity of a Geoinformation Theory"*. ISPRS Journal of Photogrammetry and Remote Sensing, 46 (1991). Elsevier Science Publishers B.V., Amsterdam, pp. 85-103.

Molenaar, M., 1994. *"A Syntax for the Representation of Fuzzy Spatial Objects"* In Molenaar, M. and S. de Hoop (eds.), *AGDM'94 Spatial Data Modelling and Query Languages for 2D and 3D Applications,* Publications on Geodesy - New Series, No. 40, pp. 155-169, Netherlands Geodetic Commission, Delft.

Molenaar, M., 1996 (c). "*An Introduction into the Theory of Topologic and Hierarchical Object Modeling in Geo-Information Systems*". Lecture Notes, Department of Land-Surveying & Remote Sensing, Wageningen Agricultural University, The Netherlands, 186 pages.

Molenaar, M., 1996 (d). "*Discrete Spatial Models for Fuzzy Geographical Objects*". In: Progress in Industrial Mathematics at ECMI 94, H. Neunzert (ed.), Wiley and Teubner, New York, pp. 454-483.

Peng, W., 1997. "*Automated Generalization in GIS*". Ph.D. Thesis, Wageningen Agricultural University, The Netherlands, 188 pages. ISBN 90 6164 1349.

Smith, J.M., and Smith, D.C.P., 1977. "*Database Abstractions: Aggregation and Generalization*". ACM Transactions on Database Systems 2, June 1977, pp. 105-133.

CHAPTER

6 The Role of Semantics in Mapping Between Database Schemas

"There isn't any symbolism. The sea is the sea. The old man is an old man. The boy is a boy and the fish is a fish. The shark are all sharks no better and no worse. All the symbolism that people say is.... What goes beyond is what you see beyond when you know."

Ernest Hemingway (1899–1961), *Letter, 13 Sept. 1952, to the critic Bernard Berenson, of The Old Man and the Sea, published that year*

6.1 Introduction

Heterogeneity is an unavoidable consequence of the design, implementation and administration autonomy of the component databases. Semantic and schematic heterogeneity are two related issues. It has been shown, in section 4.6, that to resolve schematic heterogeneity it is necessary to represent semantics in the data model. Chapter 5 presented SFDS where semantics were introduced as context information and were associated with schema elements. Several types of context information are also presented.

The objective of this chapter is to show how the semantic similarity between schema elements of a context and a proxy context can be detected. Furthermore, the chapter describes the methods to resolve the schema heterogeneity that may exist between two semantically similar schema elements.

Section 6.2 summarizes phases required for resolving the semantic and schematic heterogeneity. The first phase is to detect the semantic similarity between an export schema and the proxy context, the second is to resolve the schematic heterogeneity among the similar elements. The two phases are presented in detail in sections 6.3 and 6.4. Section 6.5 presents the steps that have to be taken to map between schemas. Before the chapter is concluded in section 6.7, a discussion on information loss in SFDS, when information is transferred from one database to another is presented in section 6.6.

6.2 The Process of Resolving Semantic Heterogeneity

As shown in Figure 4-11, the proposed approach to resolve semantic heterogeneity is based on defining an export schema for each component database and a federated schema that each export schema should map to, as well as associate context information with each of the underlying schemas. The federated schema is specific to a particular application (e.g., road network information, or relief information) and resides at the proxy context. The export schema is a description of the underlying component database, or parts of it which are to be shared. The context information is defined for all contexts and the proxy context. The database designers provide a mapping between their underlying context and the proxy context. The tasks to be performed to map between their databases and the semantic translator are the following:

1) Designing the export schema and associating its underlying context information.

2) Mapping between the federated schema of the semantic translator and the underlying context. This is achieved by:
   - Detecting the semantic similarity between the federated schema in the semantic translator and the export schema in the component database.
   - Resolving the schematic heterogeneity among the semantically similar schema elements with the aid of the context information.

The first step was the main topic of chapter 5. The second step which is the mapping between schemas is the focus of the following two sections. In section 6.3, the process of detecting the semantic similarity is presented. The types of schematic heterogeneity and the methods to resolve them are presented in section 6.4.

6.3 Detecting Semantic Similarity

The schematic heterogeneity can be reconciled only, when the semantic similarity among elements from the federated schema and the export schema are identified. Schematic conflicts can only exist among semantically similar schema elements. Detecting the semantic similarity (called schema analysis by Sheth, Gala and Navathe) requires human interaction at the design phase (at least with the current technology) [Sheth et al., 1993; Sheth, 1995]. Garcia-Solaco, Saltor and Castellanos mention that in the first generation of approaches these aspects were purely syntactic and schematic by nature, and, therefore, captured very limited semantics [Saltor et al., 1993; Garcia-Solaco et al., 1996]. Current approaches use a knowledge base (rule base, ontology, etc.) that represents concepts as a framework for semantic similarity detection.

6.3.1 Approaches to Detect Semantic Similarity

De Souza presents a model to assign weights to similar attributes. If an attribute in the export schema is related to more than one attribute in the federated schema, then the one with the highest weight is considered. Elmasri, Larson, and Navathe present the theory of attribute equivalence [Elmasri et al., 1989]. The theory compares the domains of the attributes in the export schema with the domains of the attributes in the federated schema. Attributes are semantically similar, if there exists a mapping between their domains.

The above two approaches rely on the semantic expressiveness of attributes, which is limited, if they are compared individually. In other words, taking each attribute independently does not provide complete knowledge of its semantics. For example, in DB1, attributes X and Y specify the position of a point in XY coordinates, whereas in DB2, the position is specified in polar coordinates by attributes A and R. No relationship exists between one attribute from DB1 and the other one from DB2, unless it is indicated that they are coordinates of a point. Another disadvantage is that the two approaches assume a one-to-one mapping between attributes in the underlying databases, which is not usually the case. For example, speed limit in one database is presented as an attribute of a class road, while roads can be classified according to their speed limits in another database and hence are implicitly indicated in the intension of the classes and not presented as attributes.

Bouzenhoub and Commyn-Wattiau attempt to overcome the above shortcomings [Bouzenhoub et al., 1990]. They propose a comparison between hierarchies in the export schema and the federated schema. First they compare classes from the two schemas, if they are similar then a comparison of corresponding attributes and domains is performed. This approach assumes that each class in the export schema corresponds to a class in the federated schema. This is not usually the case, for example roads can be classified according to the pavement type of the surface or according to the speed limit or their number of lanes. A road object with one surface type can have different speed limits in parts of its segments, and hence has to be decomposed.

The above approaches are developed on the assumption that the more semantic relationships exist among the object attributes, the more likely it will be that these objects are related. However, as a consequence of the limitation in the attributes to express semantics (i.e., attributes with atomic types), it is the determination of attribute relationships that constitutes the main impediment for the execution of the detection task.

6.3.2 The Adopted Approach to Detect Semantic Similarity

The approach adopted in this research is based on extending the proxy context with a list of common ontologies. Ontology is a term which is adopted from artificial intelligence, AI, to denote specification of conceptualization. Gruber defines ontology as a description (like a formal specification of a program) of the concepts and relationships that can exist for a given application or discipline [Gruber, 1993]. The concept of ontology was originally developed to support sharing and reuse of formally represented knowledge among AI systems. Kashyap and Sheth view ontology as a symbolic layer close to concepts in the real world [Kashyap et al., 1995; Sheth et al., 1993; Yu et al., 1991; Collet et al., 1991]. They define it as the specification of a representational vocabulary for a shared domain of discourse. In this sense, it is different from its use in philosophy, where it means a systematic account of existence. Ontology, as viewed in this research, can simply be thought of as the vocabulary used by experts in a certain discipline. The vocabulary is formally presented as a set of interrelated hypernyms and hyponyms.

Ontology is a hierarchy of interrelated hypernyms and hyponyms of the vocabulary that defines a shared domain.

The reason for adopting the approach of common ontology is that the mapping from schema to ontology reduces the problem of having to know the schema and semantics of databases in the large number of component information systems. It reduces the task of similarity detection to the significantly smaller problem of having to know the relationships among concepts in the ontologies. Both schemas (the federated and the export) are mapped to the common ontology. When a schema element from the federated schema is mapped to the same ontology as that of the export schema, then the two elements are semantically similar and hence the type of schema heterogeneity is identified and then resolved (more on schematic heterogeneity can be found in section 6.4).

From the practical point of view, it is suggested that a semantic translator is provided to share information of a specific application (e.g., cadastre information). The common ontology is embedded in the translator and is specific to an application, such as road network ontology, height information ontology, etc. The translator is provided, such that each element in its federated schema is pre-mapped to the common ontology.

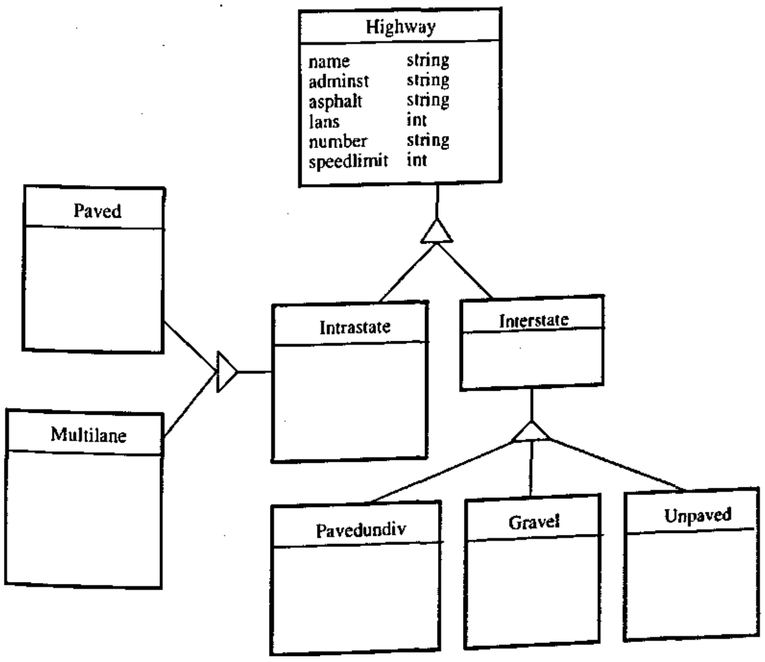


Figure 6-1 The export schema of an information resource for road information.

Example

To have some idea of the concept of detecting semantic similarity consider the federated schema for sharing road networks information shown in Figure 5-5. The common ontology is shown in Figure 6-2. The relationships among concepts are also shown (the ontology relationship is only meant to illustrate the concept and is not meant to be complete). The semantic translator is provided in such a form that the federated schema is pre-mapped to the common ontology, as shown in Table 6-1.

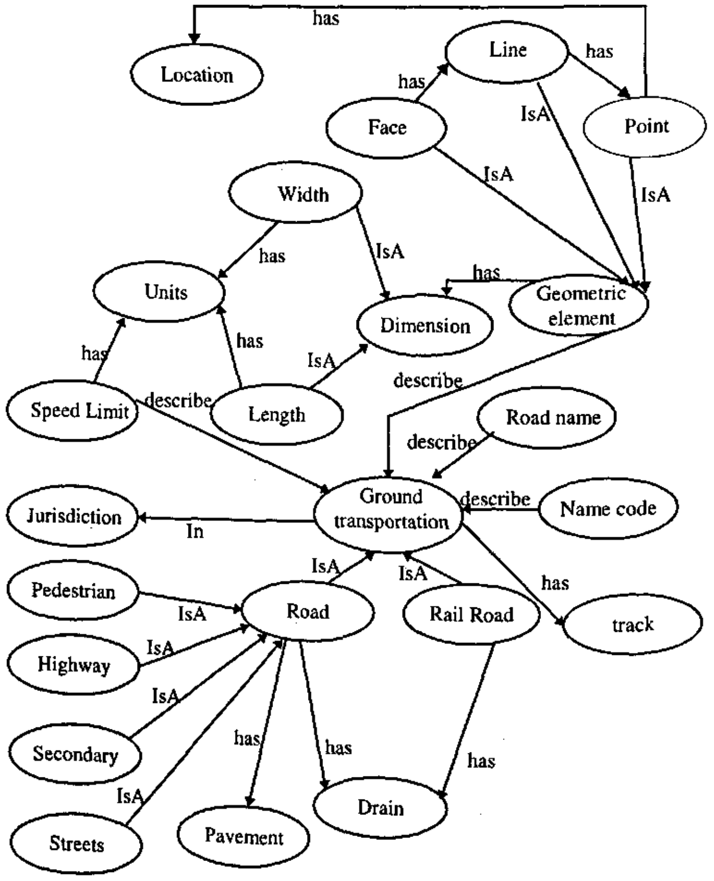


Figure 6-2 Common ontology for transportation. The ontology only represents the concepts of the domain.

Table 6-1 The federated schema is pre-mapped to the common ontology.

Federated schema	Common ontology
Pavement	Road
Highway	Highway
Street	Streets
Adminstration	Jurisdiction
Asphalt	Pavement
Speed limit	Speed limit
Geometry	Face
Number	Name code
Name	Road name
Ntracks	Track

Table 6-2 The export schema is mapped to the common ontology.

Export schema	Common ontology
Highway	Highway
Interstate	Jurisdiction
Intrastate	Jurisdiction
Gravel	Pavement
Unpaved	Pavement
Paved	Pavement
Number	Name code
Name	Road name
Speed limit	Speed limit
Geometry	Line
Lanes	Track

In the next step the database designer maps between the export schema and the common ontology. Figure 6-1 shows the export schema of a context which is to provide road information to other databases. Due to the fact that the relationships among the federated schema and the common ontology are predefined, once the relationship between the export schema and the common ontology is established, the corresponding element in the shared schema can be determined. Table 6-2 shows elements of the export schema and their corresponding common ontology.

The following step is a direct process to find the elements of the federated schema which are semantically related to the elements in the export schema, as shown in Table 6-3. The column process indicates the type of conflict among the corresponding schema elements. For instance, *unpaved* is a class in the export schema, while it is an attribute value in the federated schema. Hence, the database designer has to map a class name into an attribute value.

Table 6-3 Semantically related elements from the federated schema and the export schema, and the process required to map between them.

Export schema	Process (phase 2 see section 6.4)	Federated schema
Highway	Differences between classes	Highway
Interstate	Convert a class to attribute value	Administration
Intrastate	Convert a class to attribute value	Administration
Gravel	Convert a class to attribute value	Asphalt
Unpaved	Convert a class to attribute value	Asphalt
Paved	Convert a class to attribute value	Asphalt
Number	Map between attribute domains	Number
Name	Map between domains	Name
Speed limit	Map between domains	Domains
Lanes	Map between domains	Ntracks
Geometry	Convert boundary to centerline	Geometry

6.4 Resolving Schematic Heterogeneity

Once the semantic similarity has been detected, the next step is to resolve the schematic heterogeneity. In order to deal with heterogeneity in a systematic way, it is convenient to classify the types of schematic heterogeneity among schemes as completely as possible. In contrast to other approaches the classification adopted in this research is based on a rich object-oriented model. The reason for this is that both the export schema and the shared schema are object oriented. Kim and Seo provide an elaborate list of schema heterogeneity based on the relational database, which has been extended later to cover the object oriented model [Kim et al., 1991]. Sheth and Kashyap provide a comprehensive list of possible aspects of heterogeneity between two schemes [Sheth et al., 1992]. The following classification considers the two approaches. It is based on the aspects of the discipline world view (categories definition, class intension definition, and geometric description), mentioned in Chapter 4. Schematic heterogeneity can then be classified into four broad groups, Figure 6-3:

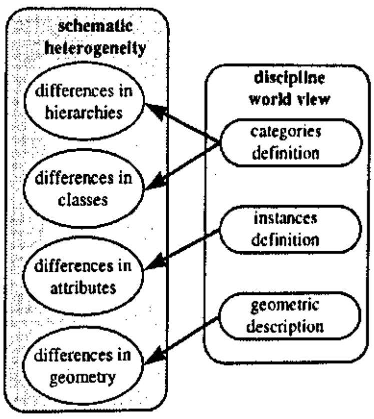


Figure 6-3 Schema heterogeneity is related to the discipline world view.

- differences in hierarchies
- differences in classes
- differences in geometry.
- differences in attribute lists and domains.

Along with the classification, methods for resolving each type will be presented. Some of these methods are also implemented in the SemWeb prototype, as will be shown in chapter 8. The description is presented under the assumption that the mapping takes place between the component databases and the semantic translator.

6.4.1 Differences in Classes

This group of differences occurs between classes defined in an export schema of a component database and those defined in the proxy context.

6.4.1.1 Synonyms and homonyms

Conflict: The term synonyms refers to the cases in which two classes, in two independent contexts, which are semantically similar have different names. The homonym problem is not considered due to the fact that two classes with the same name and representing two different concepts are not semantically similar and hence are not comparable.

Resolution: Resolving this problem requires maintaining a thesaurus that captures the correspondences among the different synonyms.

6.4.1.2 Differences in Class Attributes

Conflict: A class C may be semantically similar to another proxy class PC in the proxy context and have the same name. Yet the two classes may have discrepancies in their associated property list.

Resolution: Two classes are union compatible, if and only if they have equivalent signature. The signature need not be identical, since simple transformations such as renaming may take place. An attribute associated with C might have no correspondence in PC. In this case when the instance is copied from C to PC, the extra attribute is lost. On the other hand, when PC has an attribute which is not in C (and is not computable or does not have a default value), then it can have the value of Not Available, NA.

6.4.1.3 Differences in Methods

Conflict: Methods are functions which return values of specific type (integer, real, etc.). A class or an attribute can be semantically similar to another class or attribute in the proxy context and yet the two classes or attributes can have differences in their methods.

Resolution: Since method declaration is part of the definition of an OO class, a method can be treated just like an attribute. For example C and PC are identical except for a missing method, the method can be regarded as a missing attribute. Hence, conflicts that occur between attributes in C and PC, as mentioned in section 6.4.1.2, will apply in the case of method difference. Kim reports that when two methods have similar arguments with different types, they may be integrated by considering the data type conflicts between the arguments [Kim, 1995]. This situation may be seen as being similar to type coercion between two semantically similar attributes with two different types, as will be mentioned later in section 6.4.2.4.

6.4.1.4 Difference in Integrity Constraints

Conflicts: Two classes C and PC can be compatible in all the above aspects but still be restricted by constraints which may not be consistent with each other.

Resolution: Depending on the nature of the integrity constraints, it might be possible to generalize the constraints and have a mapping from the specific to the general constraints. For example X and PX are two semantically similar attributes. The value of X has the constraint $200 \geq x \geq 1000$, while the value of PX has the constraint $x \geq 1000$. In this case the constraint of PX is more general than that of X. However, in some cases the inconsistency may be such that the mapping may not be possible.

6.4.2 Differences in Attributes

Class attributes in a component database must be redefined and appropriately transformed, so that each attribute is compatible with that in the signature of the proxy class. Synonyms, differences in constraints, and differences in methods which were mentioned above are also applicable as types of attribute heterogeneity and hence will not be mentioned here.

6.4.2.1 Differences in Domains

Conflict: Two semantically similar attributes A and PA can be different in their domains. This type of conflict can be in different forms. The first form is known as scalar values differences, which arises when different scalar values are used to represent the same information. Another form occurs when two attributes draw values from domains with different cardinalities. Differences in cardinality might result in different precessions. For example, the domain of the attribute land-suitability in one database can be (fair, good, very good), while it is (poor, fair, good, very good) in the proxy context.

Resolution: The differences in the scalar values can be resolved by defining an isomorphism between different representations. On the other hand, when two attributes have domains with different cardinalities, it is possible to provide a mapping between a more precise domain to a less precise domain. Mapping from less precise to more precise domains is not possible if the objective is to maintain precision.

6.4.2.2 Differences in Units

Conflict: Conflicts of this type arise when numerical data denoting the same physical quantity are represented in different units (e.g., km and miles). It can also arise due to differences in the spatial reference system.

Resolution: The differences can be resolved by creating an arithmetic function which converts between units or reference systems. It is worth mentioning here that the conversion may involve loss of precision depending on the accuracy of the algorithm.

6.4.2.3 Difference in Default Values

Conflict: Conflicts of this type arise when the default value of semantically equivalent attributes are different.

Resolution: In this case a decision by the database integrator must be made to select which default value to establish.

6.4.2.4 Differences in Data Type

Conflict: As the name indicates, this type of heterogeneity occurs when, for example, an attribute in the export schema is of type real, while its corresponding semantically similar attribute in the proxy context is integer.

Resolution: In many cases it is possible to resolve this conflict by coercing the type of one attribute to another type, thus homogenizing the attributes in question. In some cases it is likely that information loss occurs, when coercing takes place from one type to the other (e.g., from real to integer).

6.4.3 Differences in Hierarchies

Here heterogeneity is classified according to differences in the hierarchies of an export schema and those of the proxy context.

6.4.3.1 Differences in Generalization

Conflict: The inconsistencies along the generalization hierarchy occur when a class C_1 in a context is semantically similar to a more general class PC_1 in the proxy context. This can be realized, when C_1 is mapped to a more general common ontology than the common ontology of the PC_1. For example, consider the common ontology shown in Figure 6-2. C_1 is mapped to *Road* while PC_1 is mapped to *Ground transportation.* In this case C_1 is included in PC_1, that is to say the extension of C_1 is a subset of the extension PC_1. Furthermore, specialization inconsistency occurs when a class C_2 is semantically similar to a more specialized class PC_2, which means that PC_2 is included in C_2.

Resolution: When the signature of class PC_1 subsumes that of C_1, then C_1 and PC_1 are said to be extended-union compatible. The notion of extended-union-compatible was defined by Date to deal with inheritance along generalization hierarchies [Date, 1995]. In the case of generalization, instances of C_1 are attached to PC_1. If there are properties in PC_1 which are not originally in C_1 then its instances (which also belong to PC_1) will have these properties and will be given the default values if they exist, or NA otherwise. On the other hand, the situation is more difficult to solve if PC_1 specializes C_1.

6.4.3.2 Differences of Aggregation Levels

Conflict: This type of inconsistency arises when the extension of a class C_1 is part of the extension of a semantically similar aggregate proxy class PC_1.

Resolution: Usually aggregation takes place at the extension level. An aggregated instance is formed by a collection of instances from one or more classes by vertical join. In the case of dis-aggregation, an instance of a class C_1 can be an aggregate and has to be decomposed into its constituent components in PC_i where $1 \leq i \leq N$. In some cases it is possible to parse the class properties and perform a clustering operation (based on the component classes) to form more than one class. This is similar to the notion of vertical join introduced by Date [Date, 1995]. Vertical join refers to aggregating objects which belong to classes of different characteristics into one complex object. For example a terrain mapping unit in a component database might be an aggregate of soil and land-use units. If the proxy context has soil and land-use classes, then a decomposition is required. The attributes of the terrain mapping units can be regrouped to form two new units (soil and land-use). However, the decomposition may not include the geometric as well as some thematic components, if they are not available from the component database. In this case the missing values can be NA.

6.4.3.3 *Class, Attribute, and Domain Differences*

Conflict: This type of conflict arises, when: 1) an attribute name in an export schema can be a class name in a proxy context; 2) the value of an attribute in an export schema corresponds to a class name in the proxy context (or vice versa).

Resolution: SFDS provides a solution based on the explicit representation of elements of the export and federated schemas. This is illustrated in the reference model, chapter 5.

6.4.3.4 *Difference in Geometry*

Conflict: as mentioned in section 5.5.2.3 differences in geometric representation of spatial objects can occur due to spatial scale differences or due to their role in their underlying context. Moreover, the four strategies of generalization and aggregation mentioned in section 5.3.2 result in a change in the geometric representation. These are, geometry-driven generalization, class-driven generalization and aggregation, functional-driven aggregation, and structural generalization.

Resolution: Several methods were proposed to resolve geometric aggregation and generalization. Richardson presents the method of rank order selection for class generalization and object aggregation [Richardson, 1993]. The approach is based on defining rules to rank objects according to their relevance at the aggregated level. Then objects are selected according to a certain threshold. Two types of rules are identified, rules which refer to the thematic content of the aggregated object and rules that refer to the geometric or topologic relationships among constituting elementary objects. Such an approach can be implemented to resolve this type of conflicts (no further reference is made to this problem in the proposed prototype).

6.5 Steps for Mapping Between Schemas

Figure 6-4 shows the process of mapping between the export schema and the federated schema. The following can be stated:

1) The design of the semantic translator includes:

   - the definition of the application to be shared,
   - the design of the common ontology,
   - the design of the federated schema,
   - mapping between the elements of the federated schema and the common ontology, and
   - defining the context information and associate it with the federated schema.

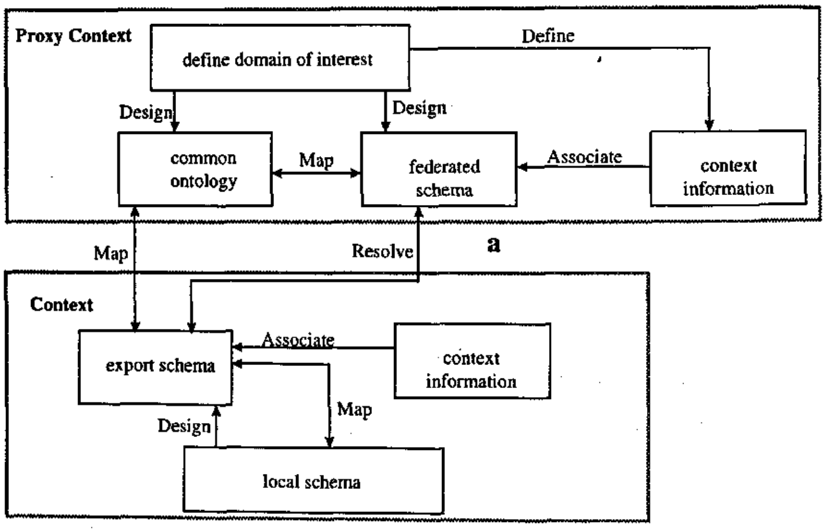
2) Preparing a context to share information includes. (This is the responsibility of the database designer)

   - design of the export schema,
   - define context information and associate it with the export schema, and
   - map between the local schema and the designed export schema.

3) Mapping between the export schema and the semantic translator includes

   - the comparison between the elements of the export schema against the common ontology,
   - find the element of the federated schema which corresponds to the identified ontology, and

- resolve the heterogeneity between the two schema elements using their context information.



b

Figure 6-4 The definition and mapping between a) Proxy Context and b) Context.

The mechanism supported by SFDS can be formulated as follows:

Let FS be the federated schema, which has the schema elements
$FS = \{fs_1, \ldots, fs_k\}$
Let CO be the common ontology which has the set of concepts
$CO = \{co_1, \ldots, co_m\}$
Each element of FS is associated with an ontology in CO, i.e.,
$(\forall fs \in FS)\ (\exists co \in CO) \Rightarrow fs \cong co$
Let ES be the export schema of a component database, which has the schema elements
$ES = \{es_1, \ldots, es_n\}$, then the semantic similarity can be
$(es \cong co) \wedge (fs \cong co) \Rightarrow (es \cong fs)$
Resolving the schematic heterogeneity starts, once the similar schema elements are defined.

Mapping between the context of an information resource (the provider) to that of the information receivers is achieved via the proxy context. The mapping may involve schema heterogeneity which can be in any of the forms shown in section 6.4. Resolving the schematic heterogeneity is achieved by functions which use context information attached to the elements of the federated schema and the export schemas.

The syntax of the function which maps the export schemas and the proxy context, was shown in section 5.5.1. After presenting the different types of schema heterogeneity, it is possible to define the set of functions which map from a schema element in a context to another semantically similar schema element in a proxy context. We then arrive at the following matrix notation.

Let C, A, M, D, G be the set of classes, attributes, methods, domains, and geometry, respectively, defined in a context Cont_i of some export schema, i.e., $\Gamma = \{C, A, M, D, G\}$. Let PC, PA, PM, PD, PG be the set of proxy classes, proxy attributes, proxy methods, proxy domains, and proxy geometry in some proxy context PCont_j, i.e., $\Gamma` = \{PC, PA, PM, PD, PG\}$.

Let

$$PCont = \left[\begin{array}{c:c} & \\ \hdashline PC & PCont_i \\ PA & PCont_j \\ PM & PCont_k \\ PD & PCont_l \\ PG & PCont_m \end{array}\right] \qquad \mathrm{Cont} = \left[\begin{array}{c:c} & \\ \hdashline C & Cont_i \\ A & Cont_j \\ M & Cont_k \\ D & Cont_l \\ G & Cont_m \end{array}\right]$$

$$\mathrm{F} = \left[\begin{array}{c:ccccc} & PC & PA & PM & PD & PG \\ \hdashline C & f_{11} & f_{12} & - & f_{14} & - \\ A & f_{21} & f_{22} & - & f_{24} & - \\ M & - & - & f_{33} & f_{34} & - \\ D & f_{41} & f_{42} & f_{43} & f_{44} & - \\ G & - & - & - & f_{54} & f_{55} \end{array}\right]$$

Then

$\Gamma` = \mathrm{F}(\mathrm{Cont}, \mathrm{PCont})$

The dashes in the above matrix notation indicate that there is no conversion between the elements. For example it is not logically possible to convert between a geometry and a class.

6.6 Information Loss in SFDS

Information loss is a result of mapping database objects from one semantic domain to another. The approach provided in SFDS attempts to minimize the semantic loss by introducing the concept of context information. Mapping the export schema to the common ontology should be done carefully. The change of semantics caused by the mapping between schemas must, therefore, be taken into consideration not only in order to decide which substitution minimizes the loss of information, but also to present to the user some kind of level of confidence in the retrieved data set. Three types of information loss are identified in this research: intensional, extensional, and geometric information loss.

1) *Intensional information loss*: The differences in classes mentioned in section 6.4.1 and the differences in hierarchies mentioned in section 6.4.3, fall under this category. This category of information loss also occurs when a class in a component database refers to a hypernym or hyponym in the common ontology. Suppose a user requests information on secondary roads and the common ontology has only the concept of roads and highway. In this case the database integrator has to decide which term the query should map to. This type of information loss also occurs at the attribute level.

2) *Extensional information loss*: The differences in the attributes mentioned in section 6.4.2 fall under this category of information loss. Extensional information loss can also be a result of the intensional information loss. It pertains to the number of instances as well as the domain of the attributes. For example when a query term is a hyponym of another term in the common ontology, it means that the corresponding class in the export schema of the user is a specialization of the semantically similar class in the federated schema. To illustrate this we shall consider the following example:

   Query: Get all Secondary Roads.
   Answer: All the highway information will be retrieved.
   The mirror situation can also occur in other cases, where the user requests road information and the information resource only provides secondary roads information.

3) *Geometric Information Loss*: This type arises due to difference in the geometric representation of the spatial objects in different databases. The design of the proxy context should minimize this type of information loss. To avoid irreversible processes (e.g. generalization) no generalization is performed by the semantic translator, instead, this is left to users' preferences.

6.7 Conclusions

The provision of an information sharing mechanism between heterogeneous information systems is based on designing a semantic translator. The semantic translator includes a federated schema described in the reference model of SFDS, context information, and an application specific common ontology hierarchy. The database designers carry out two main tasks to provide a mapping between their local databases and the semantic translator. In the first task the common ontology is used to detect the semantic similarity between the schemas involved. In the second phase the designer has to resolve the schematic similarity between the similar schema elements. The context information plays an important role in this step. It provides knowledge about the discipline world view of the proxy context and the designer's context.

As a result of this research it has to be stated that efforts to achieve complete specifications of standards for geospatial data are impossible. However, it is possible to provide sufficient specifications. It is hence necessary to develop techniques which support database evolution.

An important conclusion to be drawn from this chapter is that the concept adopted in this research promotes a dynamic approach to link databases. Updating the proxy context information or the common ontology in the semantic translator can be achieved without directly affecting the underlying database which maps to it.

References

Bouzenhoub, M., and Commyn-Wattiau, Y., 1990. *"View Integration by Semantic Unification and Transformation of Data Structures"*. Proceedings of the 9th International Conference on Entity-Relationship Approach, EPFL, Lausanne, 1990, pp. 413-430.

Collet, C., Huhns, M.N., and Shen, W.M., 1991. *"Resource Integration Using a Large Knowledge Base in Carnot"*. IEEE Computer, Vol. 24, No. 12, December 1991, pp. 55-62.

Date, C.J., 1995. *"An Introduction to Database Systems"*, Sixth edition, Addison-Wesley Publishing Company, Inc., 839 pages, ISBN 0 201 54329 X.

Elmasri, R., Larson, J., and Navathe, S., 1989. *"Schema Integration Algorithms for Federated Databases and Logical Database Design"*. Technical Report CSC-86-9, Honeywell CSDD, Minneapolis, January 1986.

Garcia-Solaco, M., Saltor, F., and Castellanos, M., 1996. *"Semantic Heterogeneity in Multidatabase Systems"*. Chapter 5 in "Object-Oriented Multidatabase Systems - A Solution for Advanced Applications" (Bukhres, O.A., and Elmagarmid, A.K., eds.). Prentice Hall, Englewood Cliffs, New Jersey 07632, pp. 129-202. ISBN 0 13 103813 2.

Gruber, T.R., 1993. *"Towards Principles for the Design of Ontologies Used for Knowledge Sharing"*. In Formal Ontology in Conceptual Analysis and Knowledge Representation (Guarino, N. and Roberto Poli, Kluwer Academic Publishers, in press.) Substantial revision of paper presented at the International Workshop on Formal Ontology, March 1993, Padova Italy. Available as Technical Report KSL 93-04, Knowledge Systems Laboratory, Standford University.

Kashyap, V., and Sheth, A., 1995. *"Semantic and Schematic Similarities Between Database Objects: A Context-based Approach"*. Appeared in The VLDB Journal, The International Journal on Very Large Data Bases, (Apers, P., Scheck, H., Gray, J., and Su., S., eds.). September 1995.

Kim., W. and Seo, J., 1991. *"Classifying Schematic and Data Heterogeneity in Multidatabase Systems"*. IEEE Computer, N0. 24, Vol. 12, December 1991, pp. 12-18.

Kim, W., 1995. *"Modern Database Systems, The Object Model, Interoperability, and Beyond"*. ACM Press, A division of the Association for Computing Machinery, Inc., Addison-Wesley Publishing, ISBN 0 201 59098 0.

Richardson, D.E., 1993. *"Automatic Spatial and Thematic Generalization Using a Context Transformation Model"*. Doctoral Dissertation, Wageningen

Agricultural University) R&B Publications, Ottawa, Canada, 1993, 149 pages.

Saltor, F., Castellanos, M.G., and Garcia-Solaco, M., 1993. "*Overcoming Schematic Discrepancies in Interoperable Databases*". In Proceedings of the IFIP WG2.6 Database Semantics Conference on Interoperable Database Systems (DS-5), Lorne, Victoria, Australia, 16-20 November, 1993. (Hsiao, D.K., Neuhold, E.J., and Sacks-Davis, R., eds.), Holland Amsterdam, pp. 191-206. ISBN 0 444 89879 4.

Sheth, A., and Kashyap, V., 1992. "*So Far (Schematically) Yet So Near (Semantically)*". (invited paper) Proceedings of the DS-5 Semantics of Interoperable Database Systems, Lorne, Australia, November 1992. (Hsiao, D.K., Neuhold, E.J., and Sacks-Davis, R., eds.) In IFIP transactions (DS-5), N.H. North-Holland Amsterdam, 1993. ISBN 0 444 89879 4.

Sheth, A., Gala, S., and Navathe, S., 1993. "*On Automatic Reasoning for Schma Integration*". International Journal of Intelligent and Cooperative Information Systems.

Sheth, A., 1995. "*Data Semantics: What, Where and How?*". Technical Report TR-CS-95-003, LSDIS Lab, Dept. of CS, University of GA, September 1995. To appear in "Database Application Semantics," Proceedings of the 6th IFIP Working Conference on Data Semantics (DS-6), (Meersman, R, and Mark, L., eds.), Chapman and Hall, London, UK, 1996.

Yu, C., Jia, B., Sun, W., and Dao, S., 1991. "*Determining Relationships among Names in Heterogeneous Databases*". SIGMOD Record, Vol. 20, No. 4., December 1991.

CHAPTER

7 The Resource Discovery Model

"There is a great satisfaction in building good tools for other people to use."

Freeman Dyson (b. 1923),
Disturbing the Universe, pt. 1, ch. 1 (1979).

7.1 Introduction

As mentioned in chapter 3, the proposed prototype, SemWeb, involves two main steps. The first is to search for the information resource in a network of data providers. The second step is to communicate with the provider via the semantic translator in order to search and retrieve the data set. The concept of semantic translators was introduced in chapters 4, 5, and 6. It was assumed that the user knows, beforehand, the location of the information resource in the network.

In this chapter the concept of resource discovery is introduced, which enables users to search for the information resource of interest. The resource discovery model, RDM, is a clearinghouse. However, it is different from other implementations in that it presents a reference model for maintaining the metadata about information resources. The search can be either made by browsing or by keyword search of the metadata in the available information resources. There are two requirements to achieve this purpose. The first is to create a server which registers all the information resources in the network. The second requirement is to structure information resources in such a way that their relationships and similarities are explicitly formulated in the schema of RDM, and therefore can be accessed and handled more efficiently. RDM provides a classification hierarchy of nodes, subnodes, and contexts. The classification is based on the similarity between the disciplines involved.

Section 7.2 provides a summary of the general characteristics of RDM and defines the different components of the metadata. The reference model of RDM is presented in section 7.3. At the end of this section an example will illustrate how the model can be implemented. Then the chapter is concluded in section 7.4. The metadata server and its implementation of RDM are described in chapter 8.

7.2 Characteristics of RDM

The resource discovery model, RDM, provides a mechanism to search for information resources. It is based on structuring the metadata of the underlying database. Metadata are data that describe the content, representation, spatial reference, quality, and administration of geographic data sets (CEN and FDGC are examples of metadata standards). Essential elements of metadata standards are:

- *Identification information*: describes the main characteristics and basic information about the data set. For example it contains, time period of content, geographic extent, source and methods of data collection.
- *Data quality information*: contains general assessment of the quality of the data set. For example it contains attribute accuracy, positional accuracy, and lineage.
- *Spatial reference information*: this section contains the description of the reference system and the means to encode the coordinates of the data set. For example it contains the horizontal and the vertical datum description, as well as the projection system used.
- *Spatial organization information*: contains information concerning the representation of the spatial information in the data set. For example, it contains information about how spatial objects are represented, e.g., vector or raster data structure.
- *Entity and attribute information*: contains information about the elements of the schema of the data set (classes and attributes). For example the information contains attribute description and type, its domain values, and its class name.

- *Distribution information*: contains information about the distributor, costs, and other restrictions on the copyright of the data set.
- *Metadata reference information*: this section is an overview of the authors of the metadata itself and the body responsible for managing the metadata.

Current implementations of the metadata merely provide documents which describe the database contents [Nebert, 1995; McLaughlin et al., 1994; Johnson et al., 1990; Alaam, 1994] (see also other clearinghouse work shown in chapter 3). Moreover, these approaches do not provide a model to maintain the consistency of the contents of the metadata server.

```
The objective of RDM is to locate data in a network
of   information   providers.   To   achieve   this
accurately  the  consistency  of  RDM  should  be
maintained.
```

Figure 7-1, shows an example of several information resources structured in a tree, so that related contexts belong to the same discipline. Two main nodes are shown, earth resources and traffic management. The earth resources node has the environmental subnode information resource, which in turn has three specialized contexts (erosion, sediment, and water quality). The traffic management node has the road network and highway patrol contexts.

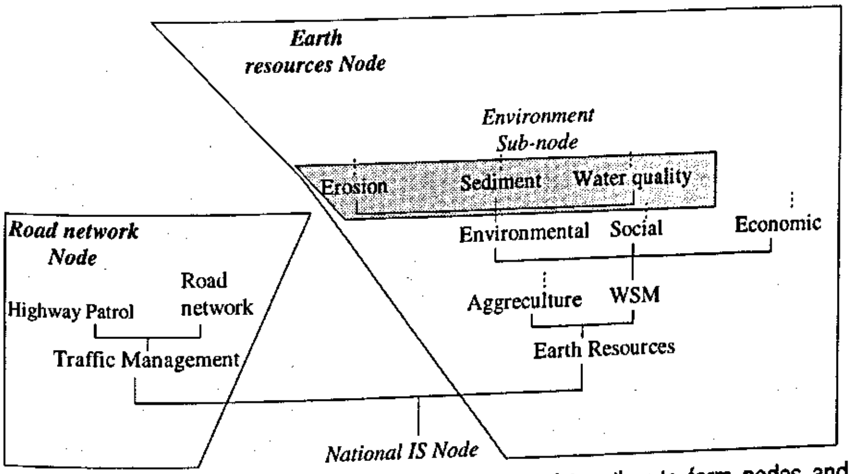


Figure 7-1 Similar application domains are grouped together to form nodes and sub-nodes.

In RDM, metadata are subsumed under four main categories:

- *General information about the information resource:* this includes information about the data provider, point of contact, spatial reference information, projection system, and time of update.
- *Metadata which describe hierarchies*: this includes type of data content, and description of the different types of hierarchies.
- *Metadata which describe classes*: cardinality, type of the geometric description of the class instances, positional accuracy, quality information, attribute list.
- *Metadata which describe attributes*: quality, type, domain, time period.

The tree of contexts has the following characteristics:

- Contexts can be structured as a tree.
- A node is a collection of related contexts.
- A node can have one or more subnodes.
- A context corresponds to one database.
- A database has one or more hierarchies.
- A hierarchy is formed by classes.
- A class has an attribute list.

7.3 Reference Model of RDM

Similar to the reference model of SFDS, which provides a template for the design of the federated schema, the reference model of RDM provides a template for the design of the tree of contexts. Elements of RDM are set in a hierarchy, as shown in Figure 7-2. Components of RDM are the nodes, subnodes, and contexts.

RDM has a tree structure, similar to the one shown in Figure 7-1. The relationship among node, subnode, context, hierarchy, class, and attribute is that of association (or member_of). A context is either a member_of a subnode or a node, the two cannot coexist. The cardinality constraints indicate that a node can have at least one subnode and one or more contexts.

Similarly, if there is a sub-node in the system, then it should be formed by one or more contexts. A class hierarchy is formed by two or more classes. If a class is a member_of a context, then the class has no structural relationship with other classes in the system i.e., it does not belong to any hierarchy.

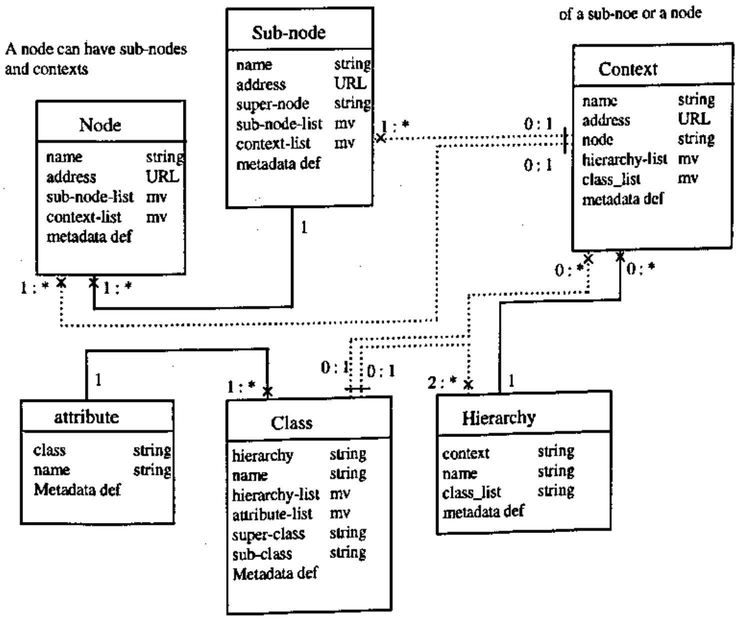


Figure 7-2 The reference model of RDM.

Two types of properties are defined in RDM. Properties which explicitly define the structure of RDM (e.g., class_list, node, and hierarchy_list) and those which explain the database itself, i.e., metadata information (e.g., positional accuracy, quality information, and attribute list). Note that metadata are presented as one attribute (that is metadata information) in the figure, however in reality they are presented as a list of attributes.

7.3.1 Integrity Constraints of the Reference Model

The integrity constraints are defined in RDM to maintain consistency between its implementations. The constraints which were introduced in section 5.3.1 are also applied to RDM and will not be mentioned here. All instances of RDM should satisfy the following conditions

- $\forall$node (sub-node-list, context-list | sub-node-list $\subseteq$ {sub-node.name} $\wedge$ context-list $\subseteq$ {context.name})
- $\forall$context (name | name $\in$ sub-node.context-list $\vee$ name $\in$ node.context-list), and
- $\forall$context (node | node $\in$ {sub-node.name} $\vee$ node $\in$ {node.name})

The following states that the set of subnode-list and context-list are subsets of the sets of sub-node.name and context.name, respectively. The context is either related to a node or to a subnode but not to both. The next assertion is a consequence of the above constraints.

> The value of context.name in any state must be an element of the node.context-list or sub-node.context-list.
> - $\forall$context (node | node $\in$ {sub-node.name} $\rightarrow$ context.name $\in$ sub-node.context-list $\wedge$ context.name $\notin$ node.context-list), or
> - $\forall$context (node | node $\in$ {node.name} $\rightarrow$ context.name $\in$ node.context-list $\wedge$ context.name $\notin$ sub-node.context-list). □

This conforms with the circumstantial membership of context, node, and subnode, shown in Figure 7-2. Now we introduce the consistency constraints between nodes and subnodes.

> $\forall$sub-node (super-node | super-node $\in$ {node.name} $\wedge$ sub-node.name $\in$ node.sub-node-list).

7.3.2 Manipulating RDM

The following definitions are intended to outline the functions which are needed to insert, update, and delete instances of RDM.

- InsertNode(name, address)
  Returns: Boolean. This function returns True, if the operation is successful and the integrity constraints are fulfilled (e.g., name and address uniqueness).

- `InsertSubnode(name, address, super-node)`
  Returns: Boolean. This function returns True, if the integrity constraints are fulfilled.

- `InsertContext(name, address, node)`
  Returns: Boolean. This function returns True, if the operation is successful and the integrity constraints are fulfilled.

- `InsertHierarchy(name, context)`
  Returns: Boolean. This function returns True, if the operation is successful and the name is unique and the context name is the same as the one the hierarchy belongs to.

- `InsertClass(name, superclass)`
  Returns: Boolean. The function returns True, if the name is unique and if superclass and subclass values are null, when the abstraction is aggregation.

- `InsertAttribute(name, type, class)`
  Return: Boolean. The function returns True, if the operation is successful and the name is unique and the value of the name is an element of the attribute-list of class.

- `Update(X, V)`
  where X is a property and V is its value.
  The function Update takes two arguments. The first argument is the property which has to be updated and the second argument is its value.

- `Delete(X)`
  where X is a property or an instance name.
  The function deletes the value of the specified property or destroys an instance, if X is an instance name.

Example

An example of the logical model of RDM is shown in Figure 7-3. The figure only shows a subset of the metadata information provided. The tables reside in the resource discovery server shown in Figure 3-5. Now suppose that we want to insert into the tables the node: traffic management and the context road network shown in Figure 7-1. The schema of the road network context is shown in Figure 6.1. Suppose also that the context has an Internet address *www.road.org*. First we insert the node.

InsertNode (traffic management, *www.road.org/index.htm*)

InsertContext (Road network, *www.road.org/~road/index.htm*, traffic management)

Here the context is inserted in such a way that it belongs to the node traffic management. Similarly, we can insert the other contexts shown in Figure 7-1. The insertion of the node and the context will trigger the integrity constraints described before.

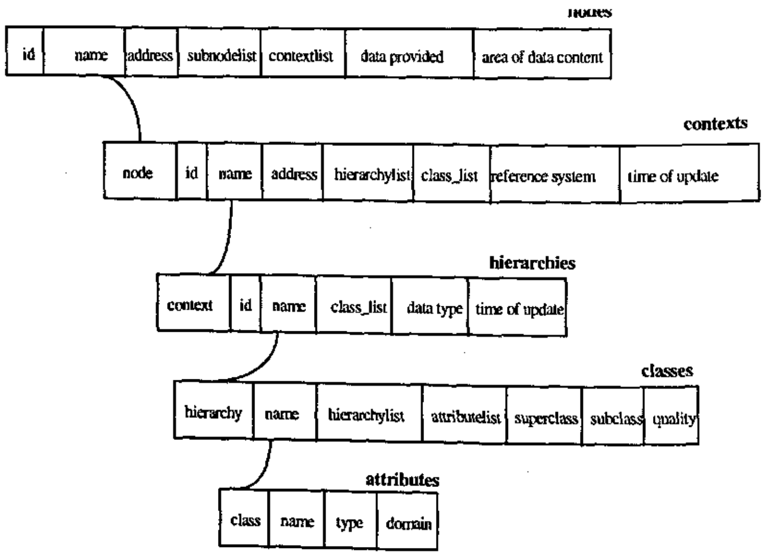


Figure 7-3 The logical model of RDM.

Next we insert the hierarchy of the road network context, this is shown in Figure 6.1. Let the hierarchy name be *freeway*

```
Insert Hierarchy (freeway, road network)
```

Then according to the class hierarchy the classes can be inserted

```
InsertClass (highway, none)
InsertClass (Intrastate, highway)
```

Similarly other classes can be inserted. When a new instance is inserted in the server, then the Update is triggered as follows:

```
If InsertAttribute then Update (Class.attribute-
list, attribute.name)
```

This condition will be triggered, once a new metadata information about an attribute is added to the server. The update will take the attribute name and update the metadata of the class attribute list. Similarly

```
If InsertClass then Update (Hierarchy.class-list,
Class.name) And Update (Context.class-list)
If InsertHierarchy then Update (Context.hierarchy-
list, hierarchy.name)
If InsertContext then Update (Node.context-list,
context.name)
If InsertSubNode then Update (Node.sub-node-list,
sub-node.name)
```

7.4 Conclusions

This chapter presents the model RDM for the resource discovery server. The model differs from current implementation of a clearinghouse in its ability to maintain the consistency of the metadata of the information resources. This provides the users with a reliable search results. As illustrated in the example, the model can be implemented in a relational database. The detailed implementation of the model, as well as the implementation of the semantic translator will be the focus of the next chapter.

References

Alaam M., 1994. "*Management Perspective of an Infrastructure for GIS Interoperability - the Delta-X Project*". Proceedings of ISPRS commission II symposium on system for data processing analysis and presentation, Ottawa, Canada. (Alaam, M., and Plunkett, G., eds.), Vol. 30 No. 2. The Survey, Mapping and Remote Sensing Sector, Natural Resource, Canada.

Johnson, D., Mott, J.J., and Musto, J.P., 1990. *"Providing Effective Access to Resource Information- Progress towards a National Directory of Australian Resource Data"*. GIS for the 1990s 532 Conference Proceedings, 1990.

McLaughlin, J. and Nichols, S., 1994. *"Developing a National Spatial Data Infrastructure"*. Journal of Surveying Engineering, Vol. 120, No. 2, May 1994.

Nebert, D., 1995. *"What does it mean to be an FGDC Clearinghouse Node?"*. Draft position paper for FDGC clearinghouse Working Group. US Geological Survey, Briston, Virginia, USA.

CHAPTER

8 Implementation of SemWeb

"The sad thing about artificial intelligence is that it lacks artifice and therefore intelligence."

Jean Baudrillard (b. 1929), *Cool Memories, ch. 4 (1987; tr. 1990).*

8.1 Introduction

In this chapter an implementation of the concepts developed in chapters 4, 5, 6, and 7 is presented. The general architecture of SemWeb is shown in Figure 3-5. SemWeb has three main components:

- The resource discovery server, which is an implementation of the resource discovery model, RDM. It allows users to search for relevant information resources. RDM is introduced in chapter 7.
- The information resource server, which has the semantic translator and the export schema of the data provider. The translator is an implementation of SFDS. It allows users to share their data seamlessly. SFDS is explained in detail in chapters 5 and 6.
- The client module which has the same semantic translator (as the one installed at the information resource) and the export schema of the client.

In this chapter SemWeb is explained in detail. The characteristics and main components of the system are introduced in section 8.1. It shows the interactions in terms of message passing among the three components. In section 8.2.1, the resource discovery server, and the tools used for its development, are explained. The information resource server and the client module are presented in section 8.2.2. The implementation of the semantic translator and the export schema of the client and the information resource are presented in section 8.3. The objective of section 8.4 is to show how the MLDSS mentioned in chapter 2 can benefit from the implementation of SemWeb. The chapter is then concluded in section 8.5.

8.2 Characteristics of SemWeb

The three components of SemWeb, the resource discovery server, the information resource server, and the client module, interact to achieve two main goals: locating the information resource that has the relevant data for the user (the client); and then retrieve the required data set by interacting with the information resource. The three components constitute the platform of SemWeb. The platform is independent of any particular application. It only provides the appropriate protocol for the three components to interact. SemWeb is extendible, new information resources can be appended to the resource discovery server and more than one semantic translator can be installed in the system. The interaction between users and SemWeb can be outlined as follows, Figure 8-1:

- Through the graphic user interface, GUI, users send the log-in request to the resource discovery server, which in turn accepts or rejects the request (depending on the access authorization).
- Using the GUI, authorized users send queries, in SQL, about the available information resources, to the resource discovery server which, in turn, searches the metadata database and sends the result back to the GUI for display. The displayed result indicates whether a particular data set exists and shows the URL of these data.
- The user can establish a link with the information resource server via the client module.
- The information resource server can accept or reject requests for connection sent by users via the client module. This depends on the access authorization.

- The data flow between the client module and the information resource server consists of queries from the user and retrieved information from the information resource.

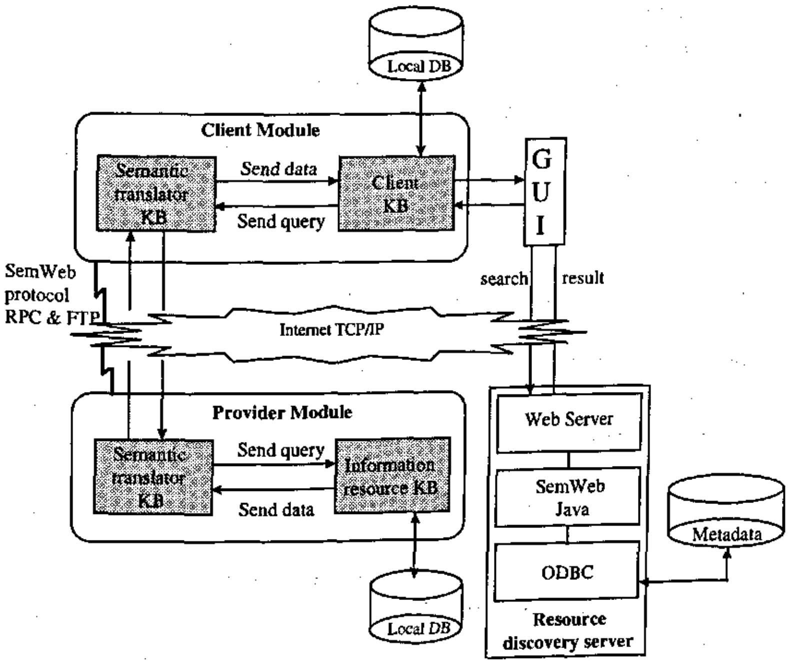


Figure 8-1 Interaction between the components of SemWeb

After introducing the general characteristics of SemWeb we shall describe its components in detail in the next two sections. Section 8.2.1 gives a detailed description of the resource discovery server and the tools used for its development. The tools used for the implementation as well as the functionality of the information resource server and the client module are explained in section 8.2.2.

8.2.1 The Resource Discovery Server

The resource discovery server employs the existing client-server technology operating on the Internet. This technology is discussed in detail in chapter 3. Figure 8-2, shows the components of the resource discovery server. The main characteristic of the resource discovery server is the structure of the metadata and hence the search procedure. In this approach the metadata are designed and implemented in a tree of nodes, contexts, hierarchies, classes, and attributes, as described in RDM. The logical model shown in Figure 7-3 is implemented in the Microsoft Access relational DBMS.

The resource discovery server has a World Wide Web server installed. The server is a software device which provides communication between the clients and the machine on which the server is installed. Java programming language is used to develop the user interface (for sending queries and receiving results) and the interaction with the DBMS. The client can run the program, using a Java capable Internet browser (e.g., Netscape or Internet Explorer), as shown in Figure 8-6. Java is a programming language to develop cross platforms portable applications, i.e., applications which can run independent of hardware and operating system.

The open database connectivity, ODBC, is used to provide the communication between the user interface (developed in Java) and the DBMS. ODBC is a standard application programming interface, API, for accessing data in relational and non-relational DBMSs. It provides the programmers with function calls to access remote databases. These functions are embedded in the Java program of the resource discovery server. The major advantage of using ODBC is that the DBMS is independent of the developed Java program which provides the GUI and the interaction between the user and the DBMS. The metadata database can virtually migrate to another DBMS without the need to change the code of the program. Users can take advantage of the SQL in querying and manipulating the database. The interface allows the user to search the metadata, using SQL statements.

As can be seen in Figure 8-2, the system allows two levels for accessing the metadata. At the first level the user interacts with global metadata which provide general information about the contents of the information resources. At the second level the user interacts with the local metadata server of the information resource itself to explore the available data further. The system functionality can be outlined as follows:

- The user runs the Java capable Internet browser and enters the Web address where the resource discovery server is located.
- The resource discovery server will run the Java program at the user's machine and display the interface.
- The user enters the query, using the SQL search conditions.

- The query is sent to the resource discovery server, where the ODBC triggers the search engine of the DBMS.
- The DBMS processes the query against the metadata.
- The server forwards the results to the *search results* panel in the GUI, as shown in Figure 8-6.
- The user can make further selections from the list of results , obtained during the previous step, till the relevant data source is found.
- The retrieved data also includes the IP address of the selected information resource.
- The user can connect to the metadata server of the information resource to explore the available data further.

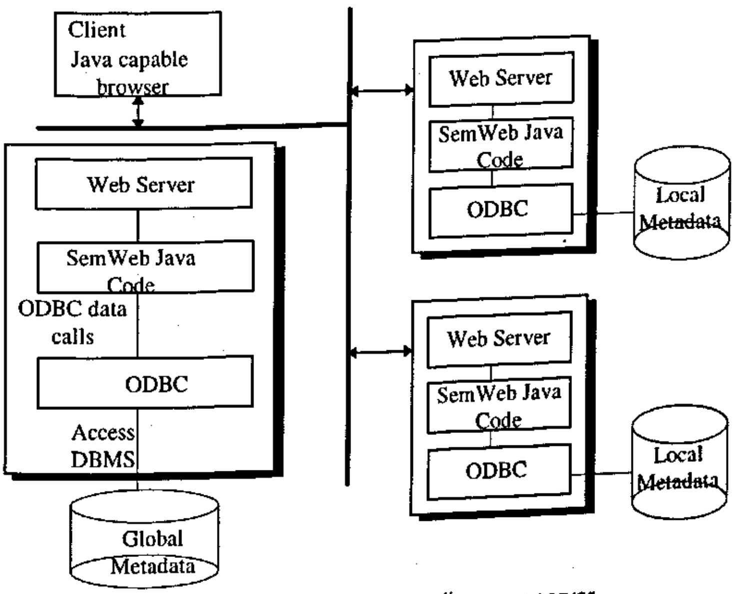


Figure 8-2 Components of the resource discovery server.

8.2.2 The Information Resource Server and the Client Module

Clients establish a link with the relevant information resource, once they have received its Internet address from the resource discovery server. The information resource server and the client module provide the necessary platform and the network protocol for running the semantic translator. The control panels of the client module and the information resource server are shown in Figure 8-7 and Figure 8-8, respectively. Although the components are two separate application programs, they are not independent. The client module cannot connect to the information resource, unless the server is running. Both application programs use Winsocket (TCP/IP) OCX controls and were developed in Visual Basic. Two main protocols are supported: the file transfer protocol, FTP, for file exchange; and the remote procedure call, RPC, for executing remote applications (e.g., the semantic translator). The two components support the following functionality:

- The client can connect to the information resource, using its IP address.

- The client can run programs installed at the local machine as well as programs installed at the machine of the information resource.

- The client can send and retrieve files from the information resource. The underlying protocol is the file transfer protocol, FTP.

Providing this interaction protocol between the information resource server and the client module makes the SemWeb platform independent of any semantic translators and export schemas. The next step is then to develop the application programs which will resolve the heterogeneity between the information resource and the client. These are the semantic translator, the client knowledge base, and the information resource knowledge base. The applications programs are then installed in the SemWeb platform.

8.3 The Implementation of SFDS

As shown in Figure 8-1, three components are installed at the client's module and the information resource server.

- The knowledge base of the information resource describes the export schema of the road information shown in Figure 8-3. Embedded in the schema is its related context information.

- The client's knowledge base describes the export schema of the road information database shown in Figure 8-4. Embedded in the schema is its related context information

- The semantic translator knowledge base describes the federated schema specific for sharing road information. The semantic translator at the client module and the information resource are similar. The federated schema is described in the reference model of SFDS and is similar to the one shown in Figure 5-5. Embedded in the schema is the related proxy context information. An example of associating context information with schema elements, as implemented in Nexpert Object, is shown later in this section.

An object-oriented expert system shell, Nexpert Object, is used to develop the aforementioned knowledge bases (an overview of Nexpert Object is presented in appendix A). These knowledge bases are embedded in a C++ program in order to automate the process of inferencing from them.

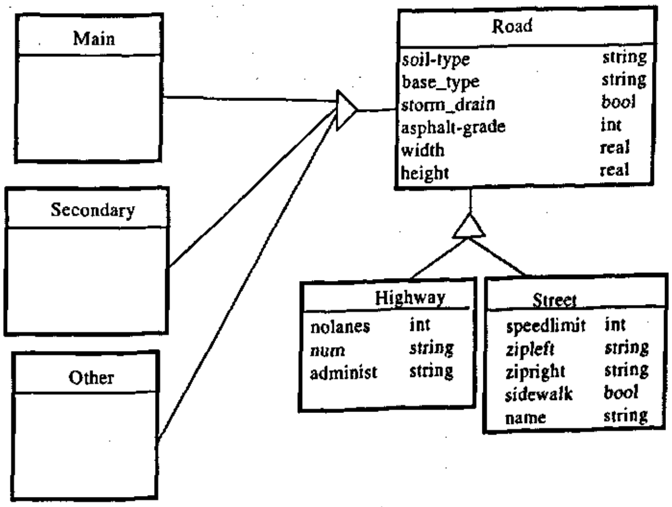


Figure 8-3 Road schema of the information resource.

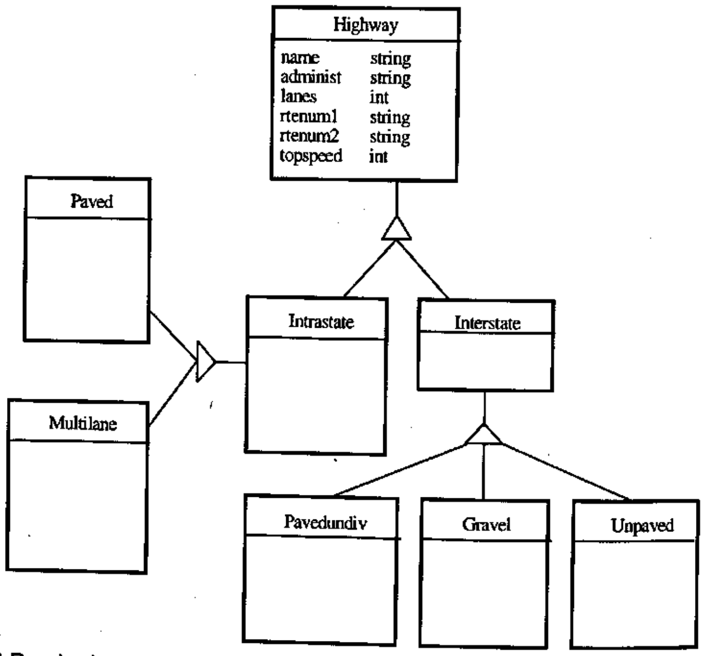


Figure 8-4 Road schema of the client.

Nexpert provides an object-oriented environment for implementing the underlying schemas. For example, a class-subclass relationship shown in Figure 8-4 can be represented in Nexpert as follows:

```
(@CLASS=          Gravel_Source
  (@PROPERTIES=
            Administrator_Source
            Fnode_Source
            Length_Source
            Lpoly_Source
            Rpoly_Source
            Tnode_ Source
            Name_Source
            Roadid_Source
            Roadno_Source
            Roadparts_Source
  )
)

(@CLASS=          Interstate_Source
  (@SUBCLASSES=
            Pavedundiv_Source
            Gravel_Source
            Multilan_Source
  )
  (@PROPERTIES=
            Administrator_Source
            Fnode_Source
            Length_Source
            Lpoly_Source
            Rpoly_Source
            TnodeSource
            Name_Source
            Roadid_Source
            Roadno_Source
            Roadparts_Source
  )
)

(@CLASS=          Intrastate_Source
  (@SUBCLASSES=
            Paved_Source
            Multilan_Source
  )
  (@PROPERTIES=
            Administrator_Source
            Maintainer_Source
            Surface_Source
  )
)
```

The following example illustrates the process of resolving a type of schematic heterogeneity between the export schema of the information resource database and the federated schema of the semantic translator. In Figure 5-5 it can be seen that the federated schema has the attribute *speedlimit*. The context information of the attribute indicates its domain {120, 100, 80} and is defined at the semantic translator. The information resource has the roads classified in such a way that the speed limits are implicitly indicated in the classification of the road network. Class *main* has the speed limit of 120 km, class *secondary* has the speed limit 80 km, etc. The problem in hand can be stated, in pseudo code, as follows:

If class *main* then *motorway.speedlimit = 120* and create new instance of *motorway* in the federated schema.

The context information associated to the classes in the export schema of the information resource indicates this type of information. The information is used to map the extension of the classes *main*, *secondary* and *others*, as attribute values in the federated schema, for which the *speedlimit* context information is taken as input. The following listing is coded at the information resource knowledge base and uses the knowledge about the domain of the attribute *speedlimit* (defined at the federated schema). The result of this process is a set of instances which are extensions of class *Motorway_can* in the federated schema, and their attribute *speedlimit* is set accordingly.

```
(@METHOD=      getrequestedobjects100_target
  (@ATOMID=Motorway_can;@TYPE=CLASS;)
  (@FLAGS=PUBLIC;)
  (@LHS=
          (=      (<<|Motorway_can|>>.Speedlimit_can)      (100))
  )
  (@RHS=
          (CreateObject    (<<|Motorway_can|>>)    (|Secondary_target|))
  )
)
(@METHOD=      getrequestedobjects120_target
  (@ATOMID=Motorway_can;@TYPE=CLASS;)
  (@FLAGS=PUBLIC;)
  (@LHS=
          (=      (<|Motorway_can|>.Speedlimit_can)(120))
  )
  (@RHS=
          (CreateObject    (<|Motorway_can|>)      (|Main_target|))
  )
)
(@METHOD=      getrequestedobjects80_target
  (@ATOMID=Motorway_can;@TYPE=CLASS;)
  (@FLAGS=PUBLIC;)
  (@LHS=
          (=      (<<<|Motorway_can|>>>.Speedlimit_can)  (80))
  )
  (@RHS=
          (CreateObject    (<<<|Motorway_can|>>>)(|Other_target|))
  )
)
(@METHOD=      getrequestedobjects_target
  (@ATOMID=Motorway_can;@TYPE=CLASS;)
  (@FLAGS=PUBLIC;)
  (@LHS=
          (=     (<|Motorway_can|>.Speedlimit_can)(120))
          (=     (<<|Motorway_can|>>.Speedlimit_can)       (100))
          (=     (<<<|Motorway_can|>>>.Speedlimit_can)  (80))
  )
  (@RHS=
          (CreateObject    (<|Motorway_can|>)        (|Main_target|))
          (CreateObject    (<<|Motorway_can|>>)    (|Secondary_target|))
          (CreateObject    (<<<|Motorway_can|>>>)(|Other_target|))
  )
)
```

Furthermore, the following types of schema heterogeneity were implemented in the semantic translator (Appendix B lists the implementation code of the knowledge bases):

- Synonyms and homonyms
- Difference in class attributes
- Difference in integrity constraints
- Difference in attributes and their domains
- Class/attribute/domain differences
- Difference in generalization and aggregation hierarchies.

The following is a description of the interaction between the client and the information resource to send queries and retrieve data:

- Using the IP address, which is returned after the resource discovery process, the client initiates a connection with the information resource server.

- A query form is displayed on the client's screen, as shown in Figure 8-9. The form assists the user to construct a query which conforms to his own semantics.

- The client KB translates the query into global terms, so that it is understood by the semantic translator. With regard to the previous example it may be observed that a client can send a query to retrieve all the instances of class *gravel* which have a *topspeed = 100*. The query is transformed at the semantic translator into retrieving the roads with *speedlimit = 100* and the *surface = 2*.

- The query is transferred from the semantic translator at the client module to the one at the information resource server.
- The query is then transferred to the information resource KB, where it is transformed into terms understood by the provider. The above query is transformed to retrieving instances of class *secondary* and *asphalt-grade = 2.*
- With the aid of the rules and methods in the information resource KB, the data set is retrieved from the underlying DMBS of the provider's GIS, which are in Dbase and ArcView, respectively. The information resource KB passes the query parameters to an AVENUE script in ArcView (AVENUE is the ArcView macro language). The script takes the query parameters and retrieves the required data set.
- The data set is transformed to the federated schema at the semantic translator and sent to the corresponding one at the client module.
- The data set is transformed from the federated schema to the export schema in the requester's knowledge base.
- The client KB runs ArcView, which in turn runs an avenue script, in order to embed the retrieved data in the underlying database as shown in Figure 8-10.

8.4 Discussion

The development of SemWeb had as its objective implementing SFDS and RDM. This is achieved by using three computers connected to a network to run the three components, the client module, the information resource server, and the resource discovery server. As mentioned in section 8.2.1, only the global metadata are implemented. However, as can be seen in Figure 8-2, users can also interact with the local metadata of the information resources to obtain more information about the available data. Furthermore, only one semantic translator is installed at the information resource and the client module. However, as mentioned in chapter 3, more than one semantic translator can be installed in SemWeb and more than one provider and client can connect to SemWeb. The implementation of SemWeb in a real case where more than one data provider, as well as metadata are involved is considered as large scale implementation.

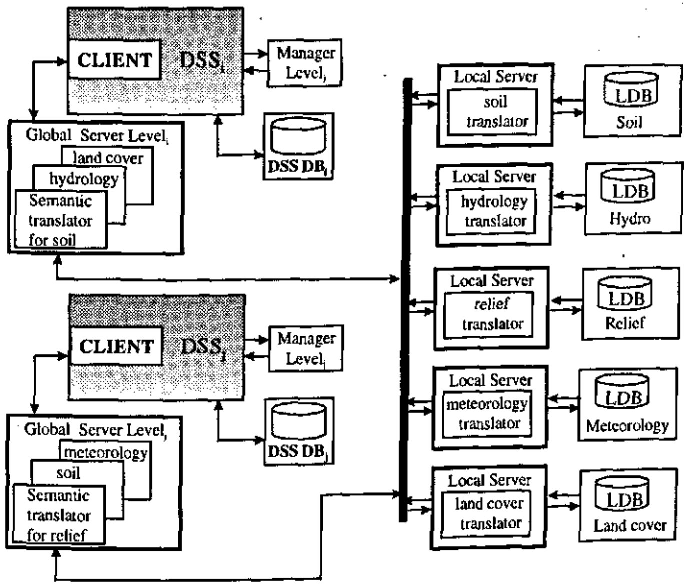

CLIENT
DSS$_i$
Manager Level$_i$
DSS DB$_i$
Global Server Level$_i$
land cover
hydrology
Semantic translator for soil
CLIENT
DSS$_j$
Manager Level$_j$
DSS DB$_j$
Global Server Level$_j$
meteorology
soil
Semantic translator for relief
Local Server soil translator
LDB Soil
Local Server hydrology translator
LDB Hydro
Local Server relief translator
LDB Relief
Local Server meteorology translator
LDB Meteorology
Local Server land cover translator
LDB Land cover

Figure 8-5 MLDSS, after the semantic translators have been installed.

The Multi-level Decision Support System, MLDSS, which was mentioned in chapter 2, is a case where large scale implementation of SemWeb is essential. Figure 8-5 shows how the semantic translators can be installed in the MLDSS. The figure only shows a part of the MLDSS. Each basic database can have a semantic translator that corresponds to its specific data type. For example, the soil and the hydrology databases have soil and hydrology semantic translators, respectively. Each DSS installs the set of semantic translators of the data types which they frequently retrieve. DSS$_i$ has the semantic translators for soil, hydrology and land cover information, while DSS$_j$ has semantic translators for meteorology, soil and relief information.

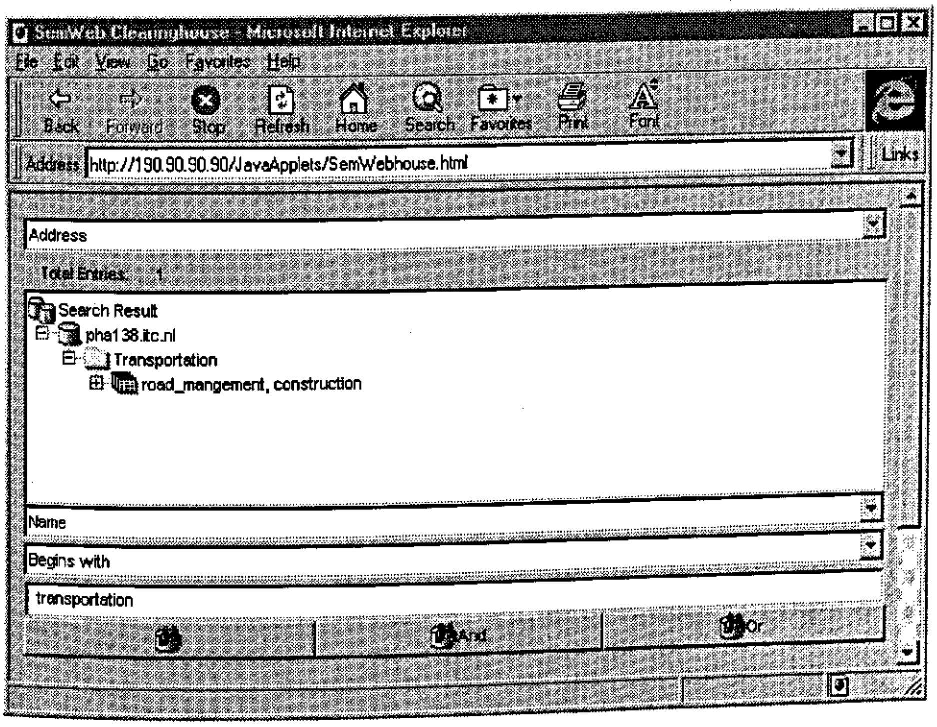
8.5 Conclusions

The resource discovery server is built, with the help of Java and ODBC. Table 1-1 summarizes the tools used for the development as well as the size of the source code. The metadata are constructed in such a way that they comply with the requirements of RDM. Furthermore, the original architecture of the resource discovery server shows several metadata databases which should reside at their corresponding information resources. They provide more detailed metadata about the underlying information resource. Only the global metadata component is implemented. Further development is still needed in this respect, where queries can be forwarded by the information discovery server to the particular information resource to explore its metadata further.

Table 8-1 Summary of the tools used for developing SemWeb

Component	Tool	Source code lines
Resource discovery server	JAVA + ODBC controls	855
Information resource server	Visual Basic 4 + Winsock OCX controls	1720
SemWeb Client Module	Visual Basic 4 + Winsock OCX controls	1345
Semantic translator	Nexpert Object + C++	400
Information resource KB	Nexpert Object + C++	415
Client KB	Nexpert Object + C++	380
Total		5115

The information resource server and the client module were developed independent of any application. The semantic translator, information resource KB and the client KB showed the applicability of implementing SFDS. Nexpert provided a proper environment to implement SFDS, where object- oriented schemas and rules are handled in a single paradigm. The queries were based on the thematic component of the spatial objects. Further development is required to provide the possibility for using spatial SQL.


SemWeb Clearinghouse - Microsoft Internet Explorer
File Edit View Go Favorites Help
Back Forward Stop Refresh Home Search Favorites Print Font
Address
Links
And
Or

Address: http://190.90.90.90/JavaApplets/SemWebhouse.html

Address

Total Entries: 1

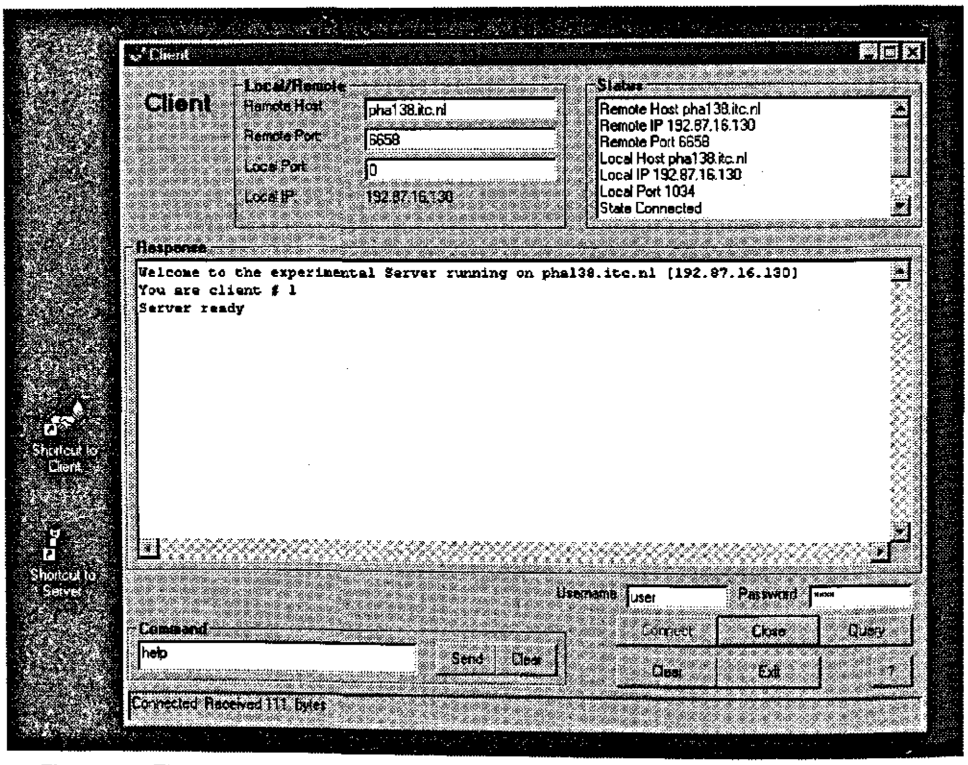
Search Result
pha138.itc.nl
Transportation
road_mangement, construction

Name

Begins with

transportation

Figure 8-6 The GUI of SemWeb clearinghouse.



Remote Host: pha138.itc.nl
Remote Port: 6658
Local Port: 0
Local IP: 192.87.16.130

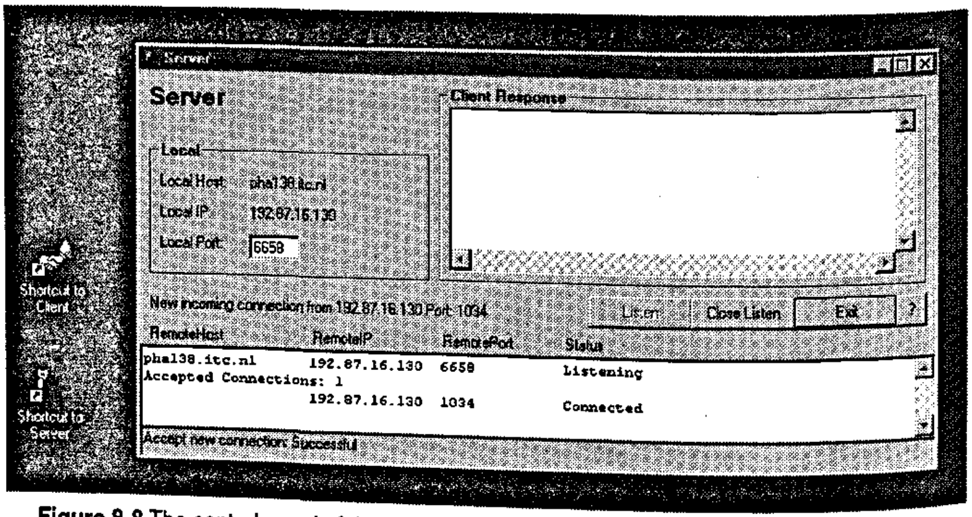
```
Remote Host pha138.itc.nl
Remote IP 192.87.16.130
Remote Port 6658
Local Host pha138.itc.nl
Local IP 192.87.16.130
Local Port 1034
State Connected
```

```
Welcome to the experimental Server running on pha138.itc.nl [192.87.16.130]
You are client # 1
Server ready
```

Username: user
Command: help

```
Connected: Received 111 bytes
```

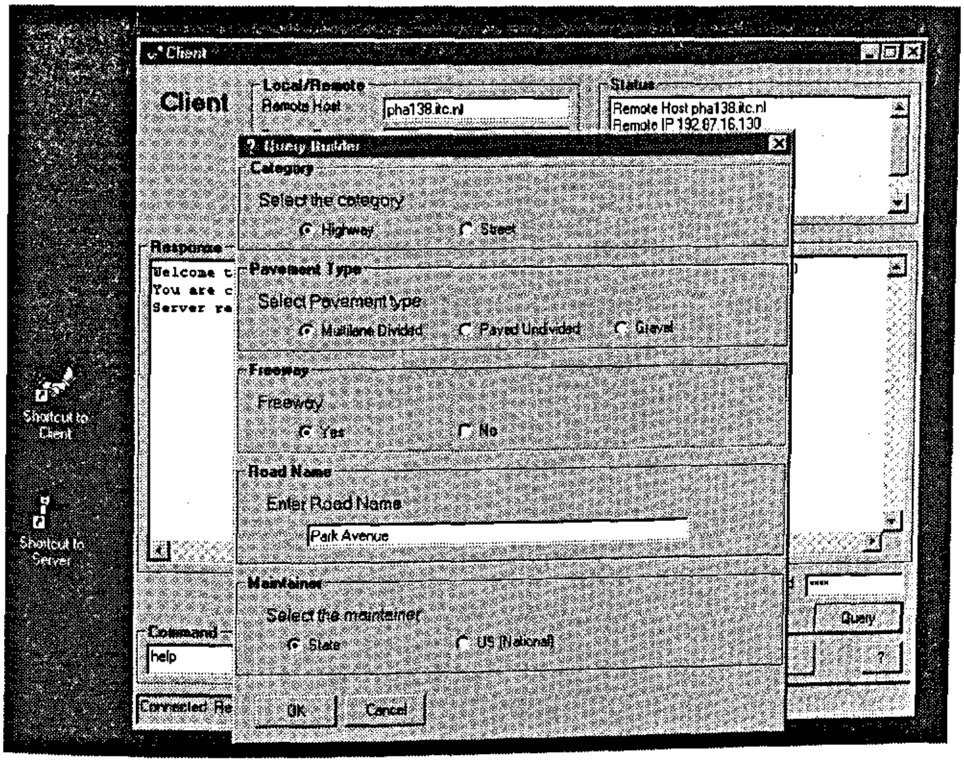
Figure 8-7 The control panel of the client module.



Local Host: pha138.itc.nl
Local IP: 192.87.16.130
Local Port: 6658

```
New incoming connection from 192.87.16.130 Port: 1034
RemoteHost       RemoteIP        RemotePort     Status
pha138.itc.nl    192.87.16.130   6658           Listening
Accepted Connections: 1
                 192.87.16.130   1034           Connected
Accept new connection: Successful
```

Figure 8-8 The control panel of the information resource server.



Remote Host: pha138.itc.nl

Remote Host pha138.itc.nl
Remote IP 192.87.16.130

```
Welcome t
You are c
Server re
```

Command: help

Connected Re

Category
Select the category
[✓ Highway] Street

Pavement Type
Select Pavement type
[✓ Multilane Divided] Paved Undivided Gravel

Freeway
Freeway
[✓ Yes] No

Road Name
Enter Road Name: Park Avenue

Maintainer
Select the maintainer
[✓ State] US (National)

Figure 8-9 Query form displayed on the client's screen.

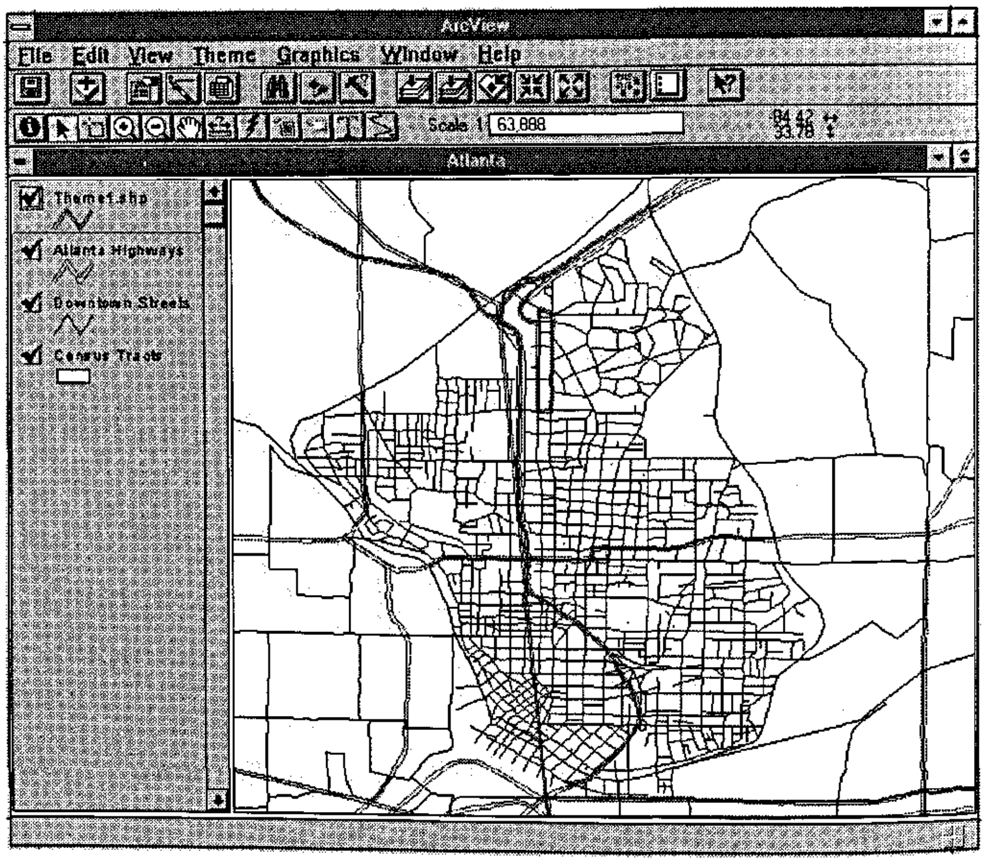


Figure 8-10 The client module automatically runs Arc/View and displays the retrieved information.

CHAPTER

9 Conclusions and Recommendations

"Just as the largest library, badly arranged, is not so useful as a very moderate one that is well arranged, so the greatest amount of knowledge, if not elaborated by our own thoughts, is worth much less than a far smaller volume that has been abundantly and repeatedly thought over."

Arthur Schopenhauer (1788–1860),
Parerga and Paralipomena, vol. 2, ch. 22, sct. 257 (1851).

9.1 Introduction

In this chapter the work described in the thesis is summarized. Conclusions are drawn about the achievements of the research together with the shortcomings and the technical considerations. Further research issues are also indicated.

9.2 Summary

The rationale of this research is to improve the process of information sharing in an MLDSS. Sharing geospatial information involves two main steps. In the first step users attempt to search for relevant information in a network of information resources. In the second step users need to retrieve information in such a way that it conforms with their data model and understanding. To make these two steps possible two models are developed in this thesis. These models are the resource discovery model, RDM, and the semantic formal data structure model, SFDS, respectively.

In chapter 1 a general overview of the problem of information sharing is provided. The chapter also looks at the information sharing problem from a wider perspective, that is, it presents the geoinformation infrastructure, GII. The research focus in only on the technical aspects of the GII

Chapter 2 discusses the problems related to environmental decision making from the geo-information perspective. The intention of the chapter is not to provide a detailed discussion of the process of environmental decision making, rather, it shows the role of geo-informatics, which is realized during a field study and an experimental implementation of an MLDSS. The chapter concludes with a list of guidelines which are observed during the research.

Chapter 3 provides a general overview of the state-of-the-art technology for interoperability. The overview organizes interoperability into six levels: network protocols, file systems, remote procedure calls, search and access databases, GIS, and application interoperability. It is shown that semantic translators are related to the interoperability at the application level. A general architecture of the SemWeb prototype, which is developed in this research, is also presented.

In chapter 4, three types of heterogeneity are defined and explained in detail: syntactic heterogeneity, schematic heterogeneity and semantic heterogeneity. Chapter 4 briefly introduces the semantic formal data structure, SFDS, to resolve the heterogeneity problem.

SFDS is explained in detail in chapter 5. It is shown that SFDS is based on providing a proxy context for information sharing. A proxy context is a mediator for sharing information between two or more GISs. It defines a common UoD, which is the domain where two or more independent databases share their data. SFDS has three layers. At the first layer FDS is adopted to resolve the syntactic heterogeneity. At the second layer a reference model for federated schemas is developed to resolve the schematic heterogeneity. At the third layer context information is represented as first order predicates and associated with schema elements of the federated schema to resolve the semantic heterogeneity.

The mapping between the schema elements of the federated schema and that of the provider, or the client, makes it necessary first to identify semantically similar elements. In chapter 6 the notion of common ontology is introduced to meet such a requirement. An element of the federated schema is semantically similar to an element in the underlying database, if they both map to the same common ontology. After the identification of the semantically related schema elements the next step is to resolve the schematic heterogeneity. A stepwise procedure to map between the federated schema of the semantic translator and the schema of the underlying database is also explained.

RDM is introduced in chapter 7. Elements of RDM are nodes, subnodes, context, hierarchies, classes, and attributes. Nodes and subnodes are collections of related contexts which belong to the same discipline. Contexts consist of hierarchies which, in turn, consist of classes and attributes. Metadata of each of these elements are set up in a database for users to search.

On the basis of the system architecture of SemWeb, SFDS and RDM are tested in chapter 8. The prototype has three components:

- The resource discovery server, which is an implementation of RDM.
- The client module, which has the export schema of the underlying database and the semantic translator, which is an implementation of SFDS.
- The information resource server, which has the export schema of the underlying databases of the provider and the semantic translator, which is an implementation of SFDS.

9.3 Discussion

Current research and industrial efforts to provide interoperable GIS mainly focus on platforms interoperability. This thesis tackles an area of research which is at a higher level than platforms interoperability. That is to say interoperability between geospatial applications.

Addressing the problem of heterogeneity requires a classification of the types of heterogeneity. Perhaps the power of SFDS as a model to provide data sharing, stems from the coherent classification defined in this research. SFDS has an advantage over the context mediation and federated database systems approaches (refer to table 4-1 for a detailed comparison). SFDS has a good support for context mediation, as well as the mapping between database schemas. Introducing common ontology can greatly simplify the process of finding semantically similar schema elements.

Implementing SFDS and RDM in SemWeb shows that the two models can mutually complement each other. Hence, unlike the context mediation and federated databases systems approaches, RDM provides a model for resource discovery. The model is basically a clearinghouse. However, it is different from other approaches in that it provides a reference model to structure metadata. This provides more intuitive, efficient, and reliable search procedures for information resources.

The limitation of SFDS is twofold. Firstly, it does not automatically recognize the semantic similarity between elements of the federated schema and the export schema. Secondly, it does not support distributed query optimization. Users can only query one information resource at a time.

9.4 Conclusions

From the various issues addressed in this work the following conclusions can be drawn:

- The development of RDM and its implementation shows that it is essential to have a consistent model for geospatial clearinghouses.
- Attempts to provide complete standards and their profiles, for representing geospatial information in the database are not successful. This is due to different users' requirements and the demand made by users to have autonomy in designing and managing their databases.
- Dynamic approaches for information sharing, as provided by semantic translators, are more powerful than the current approaches which promote standards.
- Multi-level decision support systems, MLDSSs, for environmental decision making can greatly benefit from application domain interoperability.
- Associating context information to schema elements provides a dynamic mapping between heterogeneous schemas.
- Semantic translators do not ensure information sharing without information loss, rather they attempt to minimize it by enriching the database schema with context information.
- The semantic formal data structure, SFDS, provides a framework for designing semantic translators which are considered as vehicles for achieving interoperability between geospatial applications.

- It is essential to find semantically similar elements, from the export schemas of the clients or the information resources, and the semantic translator, before resolving schematic heterogeneity.
- Ontology provides a sufficient reference to find semantically similar schema elements.
- There does not exist a comprehensive library of common ontological definitions for GIS application domains.
- Knowledge base expert systems have the proper capabilities to develop semantic translators.
- Semantic translators can be provided in class libraries, which can be linked to the export schemas, where database designers can access them to provide mapping between them and their export schemas.

9.5 Future Research

A three-layered semantic formal data structure, SFDS, is proposed. The formal data structure, FDS, forms the first layer of SFDS. The schematic and the semantic layers provide the essential details to resolve heterogeneity between databases. The resource discovery model, RDM, is introduced as consistent model for clearinghouses. Several issues are identified which will need further research and development.

- A complete formalism for semantic translators will be required. The formalism should provide a mathematically sound reference model as well as the functions to handle semantic translators.
- It will be necessary to explore further the issue of heterogeneity specific to geometric representation and quality of geospatial objects retrieved from different sources, as well as methods to resolve them.
- The research assumes that the queries sent by clients can be processed at one information resource. Further investigation into parallel and distributed query optimization for geospatial databases will be required.
- It will be necessary to develop mechanisms that automatically identify semantic similarities and resolve schematic heterogeneity. This will create more dynamic interaction between clients and information providers.
- The capability of information sharing needs to be extended to share services (e.g., GIS analysis capabilities on the Internet). Techniques for providing remote GIS services as well as distributed spatial data processing will be needed.

- Development of common ontologies for different application domains will be required.
- Generic federated schemas for different application domains will be needed.
- Research for developing semantic translators for different geospatial applications will be required.
- Spatial data are usually counted in megabytes. Efficient techniques for data compression, before these data are sent across networks, will be required.
- Organizational and economic issues related to information sharing are areas which will require more attention.

APPENDIX

A Nexpert Object

A.1 The Semantic Translator

Nexpert is a hybrid system which has an inference engine, and an object-oriented shell. Figure A-1 represents the different dimensions of Nexpert Object. The horizontal plane contains the rule base, while the vertical plane contains the object structures. The point on the intersection line illustrates the integration of rules and objects. These two main components, objects and rules, are transparent to each other in Nexpert, in the sense that, a change in an object status may trigger the inference mechanism to evaluate a hypothesis or a set of hypotheses. A rule evaluation may reflect a change in the object status.

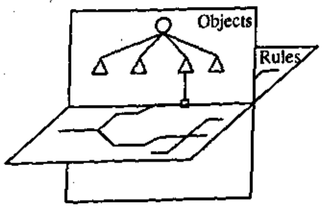


Figure A-1 Dimensions of Nexpert Object.

A.2 Rule-Based Nexpert System Shell

Nexpert provides a powerful inference mechanism for rules which are necessary to capture knowledge required for the solution of a specific problem domain. Rules in Nexpert have three Boolean basic parts, as shown in Figure A-2:

- Left-hand side conditions,
- The hypothesis
- The right-hand side actions.

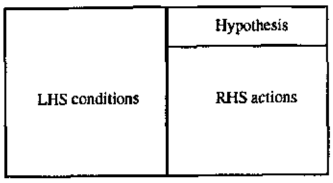


Figure A-2 Rules in Nexpert.

Conditions represent the "IF" clause which may have series of tests to evaluate the status of hypotheses (True of False). If all the conditions are True, then the hypothesis is set to True and the right-hand side actions are all executed. An Else statement can be formulated, which is executed when the hypothesis is evaluated to False.

The evaluation of rules is made through a prioritized queue called the agenda. It contains those rules which the system believes have positive evidence for being relevant to the problem under focus. The agenda can re-prioritize the relevant rules, if a new rule is inserted according to the applied search method.

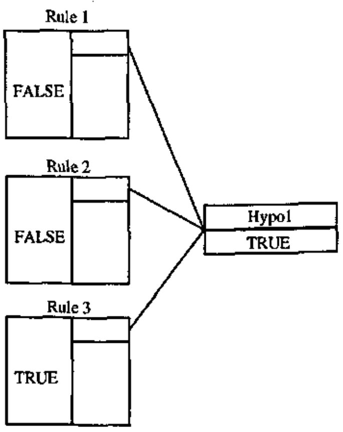


Figure A-3 Multiple Rules Evaluation

Multiple Rules Evaluation: is the process of evaluating different rules leading to the same hypotheses as shown in Figure A-3. At least one rule must be true for the hypotheses to be true. Intuitively, all rules must be false in order for the hypotheses to be false. This means that conditions within the body of the same rule are connected with 'and' while rules with the same hypotheses are connected with 'or'.

Inference Mechanism: One of the main advantages of the inference engine is its ability to expand the search space for relevant conclusions, without exhaustively evaluating the whole space of the knowledge base. Putting rules in the agenda is determined by seven types of search mechanisms; they are mentioned according to their priority in the agenda, as shown in Figure A-4:

- *Backward Chaining:* If a condition has a slot value unknown (which is in fact a hypothesis), then the rules pointing to that hypothesis will be evaluated immediately.
- *Suggesting:* If a hypothesis is suggested interactively (from the pull down menu), or dynamically (with the "control_session" routine), it means that the hypothesis is an important goal and should be evaluated.

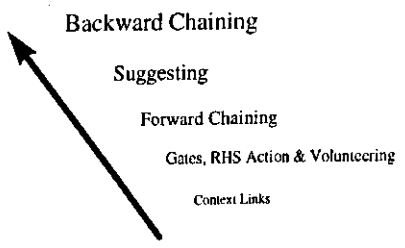


Figure A-4 Inferencing Priorities in Nexpert.

- *Foreword Chaining:* This type of hypothesis is put on the agenda due to hypothesis forward events. Hypothesis forward is a consequence of investigating sub-goals as opposed to a terminal hypothesis.
- *Gates:* This is the basic mechanism for the automated goal evaluation and opportunistic reasoning. When a slot value is determined in the LHS part of a rule, Nexpert searches for rules sharing the same slot in its LHS and puts it in the agenda for later evaluation.
- *Context Links:* Have the lowest level of priority in the agenda. They are used to link two knowledge islands through the context link editor. Knowledge islands are sets of LHS conditions and hypotheses linked with each other, at run-time, via backward chaining, forward chaining, or gates. Context links are used, if there is no way to propagate control from one knowledge island to another.

A.3 Nexpert Object-Oriented Shell

Nexpert provides a powerful object-oriented shell, where almost all the concepts can be implemented. Classes, sub-classes, objects, inheritance from parents down to children, aggregation, association, generalization, methods applied to objects, and constraints, can be presented.

Generalization is represented by a class which is connected to many sub-classes. These sub-classes are considered as specializations of the parent class. If an object has many sub-objects, at the object level, then it is considered as an aggregated or associated object (depending to the underlying case).

Methods in Nexpert can be defined for attributes, objects, or classes. In this sense, methods are encapsulated with the definition of attributes, objects, or classes. Methods describe the behavior of individual attributes, objects, or classes. Methods are composed primarily of a set of actions, which when executed modify the behavior of the object upon which they act.

APPENDIX

B The Knowledge Bases

B.1 The Semantic Translator

The following is a list of the code of the semantic translator for road information. It represents the federated schema and context information of the proxy context. The code looks like C and is understood by Nexpert. Note that the translator has to be loaded with either the client knowledge base or the provider knowledge base.

```
(@VERSION=      030)
(@PROPERTY=     adminstrator       @TYPE=String;)
(@PROPERTY=     Adminstrator_can   @TYPE=String;)
(@PROPERTY=     class1_can         @TYPE=String;)
(@PROPERTY=     Direction_can      @TYPE=String;)
(@PROPERTY=     Expressway_can     @TYPE=Boolean;)
(@PROPERTY=     fnode_can          @TYPE=Integer;)
(@PROPERTY=     length_can         @TYPE=Float;)
(@PROPERTY=     lpoly_can          @TYPE=Integer;)
(@PROPERTY=     Name_can           @TYPE=String;)
(@PROPERTY=     Prop1_can          @TYPE=String;)
(@PROPERTY=     Prop2_can          @TYPE=String;)
(@PROPERTY=     Prop3_can          @TYPE=String;)
(@PROPERTY=     Prop4_can          @TYPE=String;)
(@PROPERTY=     Roadid_can         @TYPE=Integer;)
(@PROPERTY=     Roadname_can       @TYPE=String;)
(@PROPERTY=     Roadparts_can      @TYPE=Integer;)
(@PROPERTY=     Roadtype_can       @TYPE=String;)
(@PROPERTY=     roadtype_target    @TYPE=String;)
(@PROPERTY=     Route1_can         @TYPE=String;)
```

```
(@PROPERTY=     rpoly_can        @TYPE=Integer;)
(@PROPERTY=     Speedlimit_can   @TYPE=Integer;)
(@PROPERTY=     tnode_can        @TYPE=Integer;)
(@PROPERTY=     Zipleft_can      @TYPE=Integer;)
(@PROPERTY=     ZipRight_can     @TYPE=Integer;)


(@CLASS=  Highway_can
    (@PROPERTIES=
            Adminstrator_can
            Expressway_can
            fnode_can
            length_can
            lpoly_can
            Roadid_can
            Roadname_can
            Roadparts_can
            Roadtype_can
            Route1_can
            rpoly_can
            Speedlimit_can
            tnode_can
    )
)

(@CLASS=  Query_can
    (@PROPERTIES=
            class1_can
            Prop1_can
            Prop2_can
            Prop3_can
            Prop4_can
    )
)

(@CLASS=  Roadatlanta_can
    (@SUBCLASSES=
            Streets_can
            Highway_can
    )
    (@PROPERTIES=
            fnode_can
            length_can
            lpoly_can
            Roadparts_can
```

```
                rpoly_can
                tnode_can
        )
)

(@CLASS=  Streets_can
        (@PROPERTIES=
                Direction_can
                fnode_can
                length_can
                lpoly_can
                Name_can
                Roadparts_can
                rpoly_can
                tnode_can
                Zipleft_can
                ZipRight_can
        )
)


(@OBJECT=aggregate
        (@PROPERTIES=
                Value     @TYPE=Boolean;
        )
)

(@OBJECT=writeobjects
        (@PROPERTIES=
                Value     @TYPE=Boolean;
        )
)
(@METHOD=          loadhighwayobjects_can
        (@ATOMID=Highway_can;@TYPE=CLASS;)
        (@FLAGS=PUBLIC;)
        (@RHS=
                (Retrieve("C:\proto\request.dbf")
        (@TYPE=DBF3;@FILL=ADD;@NAME="!name!";@CREATE=|High
way_can|;\
@PROPS=Expressway_can,Roadtype_can,Route1_can,\
Speedlimit_can,Roadid_can,Speedlimit_can,\
Adminstrator_can,Roadname_can,Roadparts_can,\
length_can;@FIELDS="expressway","roadtype",\
"route1","speedlimit","roadid","speedlimit",\
"admnclass","roadname","roadparts","length";))
```

```
    )
)
(@METHOD=         setsimilarprops_can
    (@ATOMID=Highway_can;@TYPE=CLASS;)
    (@FLAGS=PUBLIC;)
    (@RHS=
(Assign        (SELF.Fnode_source)        (SELF.fnode_can))
(Assign        (SELF.Tnode_source)        (SELF.tnode_can))
(Assign        (SELF.Rpoly_source)        (SELF.rpoly_can))
               (Assign  (SELF.Lpoly_source)        (SELF.lpoly_can))
               (Assign  (SELF.Adminstrator_Source)
    (SELF.Adminstrator_can))
               (Assign  (SELF.Roadid_source)       (SELF.Roadid_can))
(Assign        (SELF.roadname_source)    (SELF.Roadname_can))
               (Assign  (SELF.Name_Source)         (SELF.Roadname_can))
(Assign        (SELF.roadparts_source)   (SELF.Roadparts_can))
(Assign        (SELF.Length_Source)      (SELF.length_can))
    )
)
(@METHOD=         setsimilarprops_target
    (@ATOMID=Highway_target;@TYPE=CLASS;)
    (@FLAGS=PUBLIC;)
    (@RHS=
(Assign        (SELF.Expressway_can)     (SELF.freeway_target))
               (Assign  (SELF.Adminstrator_can)
    (SELF.Maintainer_target))
               (Assign  (SELF.Roadid_can)          (SELF.ID_target))
               (Assign  (SELF.Roadname_can)        (SELF.Name_target))
               (Assign  (SELF.Route1_can)          (SELF.route1_target))
               (Assign  (SELF.Roadparts_can)
    (SELF.Roadcomp_target))
(Assign        (SELF.length_can)          (SELF.Length_target))
               (Assign  (SELF.Roadtype_can)
    (SELF.Asphaltgrade_target))
               (Assign  (SELF.Speedlimit_can)      (SELF.Speed_target))
    )
)
(@METHOD=         writeobject_can
    (@ATOMID=dumyclass;@TYPE=CLASS;)
    (@FLAGS=PUBLIC;)
    (@RHS=
               (Write   ("C:\proto\request.dbf")
    (@TYPE=DBF3;@FILL=NEW;@PROPS=Expressway_can,\
Roadtype_can,Route1_can,Speedlimit_can,fnode_can,\
tnode_can,lpoly_can,rpoly_can,Roadid_can,\
Adminstrator_can,Roadname_can,Roadparts_can,\
length_can;@FIELDS="expressway","roadtype",\
"route1","speedlimit","fnode_","tnode_","lpoly_",\
"rpoly_","roadid","admnclass","roadname",\
"roadparts","length";@ATOMS=<ldumyclassl>;))
    )
)
```

B.2 The Client Module KB

Following is a list of the code of the client knowledge base. It represents the export schema and context information of the client.

```
(@VERSION=      030)
(@PROPERTY=     Asphaltgrade_target        @TYPE=String;)
(@PROPERTY=     Class1_target      @TYPE=String;)
(@PROPERTY=     Class2_target      @TYPE=String;)
(@PROPERTY=     Class3_target      @TYPE=String;)
(@PROPERTY=     Direction_target   @TYPE=String;)
(@PROPERTY=     dum1     @TYPE=String;)
(@PROPERTY=     expressway_target @TYPE=Boolean;)
(@PROPERTY=     freeway_target     @TYPE=Boolean;)
(@PROPERTY=     ID_target          @TYPE=Float;)
(@PROPERTY=     Length_target      @TYPE=Float;)
(@PROPERTY=     Maintainer_target @TYPE=String;)
(@PROPERTY=     Name_target        @TYPE=String;)
(@PROPERTY=     Nolans_target      @TYPE=Integer;)
(@PROPERTY=     Prop1_target       @TYPE=String;)
(@PROPERTY=     Prop2_target       @TYPE=String;)
(@PROPERTY=     Prop3_target       @TYPE=String;)
(@PROPERTY=     Prop4_target       @TYPE=String;)
(@PROPERTY=     Roadcomp_target    @TYPE=Float;)
(@PROPERTY=     Roadname_query     @TYPE=String;)
(@PROPERTY=     route1_target      @TYPE=String;)
(@PROPERTY=     Sidewalk_target    @TYPE=Boolean;)
(@PROPERTY=     Signal_target      @TYPE=Boolean;)
(@PROPERTY=     Speed_target       @TYPE=Float;)
(@PROPERTY=     Zipl_target        @TYPE=Integer;)
(@PROPERTY=     Zipr_target        @TYPE=Integer;)


(@CLASS=  Highway_target
    (@SUBCLASSES=
            Main_target
            Secondary_target
```

```
                Other_target
        )
        (@PROPERTIES=
                Asphaltgrade_target
freeway_target
                ID_target
                Length_target
                Maintainer_target
                Name_target
                Nolans_target
                Roadcomp_target
                route1_target
                Speed_target
        )
)

(@CLASS=  Main_target
        (@PROPERTIES=
                Asphaltgrade_target
freeway_target
                ID_target
                Length_target
                Maintainer_target
                Name_target
                Nolans_target
                Roadcomp_target
                route1_target
                Speed_target
        )
)

(@CLASS=  Other_target
        (@PROPERTIES=
                Asphaltgrade_target
                freeway_target
                ID_target
                Length_target
                Maintainer_target
                Name_target
                Nolans_target
                Roadcomp_target
                route1_target
                Speed_target
        )
)
```

```
(@CLASS=  Query_target
     (@PROPERTIES=
               Class1_target
               Class2_target
               Class3_target
               Prop1_target
               Prop2_target
               Prop3_target
               Prop4_target
     )
)

(@CLASS=  Road_target
     (@SUBCLASSES=
               Street_target
               Highway_target
     )
     (@PROPERTIES=
               Asphaltgrade_target
               ID_target
               Length_target
               Name_target
               Roadcomp_target
     )
)

(@CLASS=  Secondary_target
     (@PROPERTIES=
               Asphaltgrade_target
freeway_target
               ID_target
               Length_target
               Maintainer_target
               Name_target
               Nolans_target
               Roadcomp_target
               route1_target
               Speed_target
     )
)

(@CLASS=  Street_target
     (@PROPERTIES=
               Asphaltgrade_target
```

```
                Direction_target
                ID_target
                Length_target
                Name_target
                Roadcomp_target
                Sidewalk_target
                Signal_target
                Zipl_target
                Zipr_target
        )
)


(@OBJECT=dumy
        (@PROPERTIES=
                dum1
        )
)

(@OBJECT=loadreceivedobjects
        (@PROPERTIES=
                Value     @TYPE=Boolean;
        )
)

(@OBJECT=Send_query
        (@PROPERTIES=
                Value     @TYPE=Boolean;
        )
)
(@METHOD=          getrequestedobjects100_target
        (@ATOMID=Highway_can;@TYPE=CLASS;)
        (@FLAGS=PUBLIC;)
        (@LHS=
                (=        (<<|Highway_can|>>.Speedlimit_can)          (100))
        )
        (@RHS=
                (CreateObject      (<<|Highway_can|>>)
        (|Secondary_target|))
        )
)
(@METHOD=          getrequestedobjects120_target
        (@ATOMID=Highway_can;@TYPE=CLASS;)
        (@FLAGS=PUBLIC;)
        (@LHS=
                (=        (<|Highway_can|>.Speedlimit_can)   (120))
        )
        (@RHS=
                (CreateObject     (<|Highway_can|>)          (|Main_target|))
        )
)
(@METHOD=         getrequestedobjects80_target
        (@ATOMID=Highway_can;@TYPE=CLASS;)
        (@FLAGS=PUBLIC;)
        (@LHS=
                (=        (<<<|Highway_can|>>>.Speedlimit_can)       (80))
        )
        (@RHS=
                (CreateObject     (<<<|Highway_can|>>>)    (|Other_target|))
        )
)
(@METHOD=         getrequestedobjects_target
        (@ATOMID=Highway_can;@TYPE=CLASS;)
        (@FLAGS=PUBLIC;)
        (@LHS=
                (=        (<|Highway_can|>.Speedlimit_can)   (120))
                (=        (<<|Highway_can|>>.Speedlimit_can)         (100))
                (=        (<<<|Highway_can|>>>.Speedlimit_can)       (80))
        )
        (@RHS=
                (CreateObject     (<|Highway_can|>)          (|Main_target|))
                (CreateObject     (<<|Highway_can|>>)
        (|Secondary_target|))
                (CreateObject     (<<<|Highway_can|>>>)    (|Other_target|))
        )
)
(@METHOD=         writeobjects_target
        (@ATOMID=Highway_target;@TYPE=CLASS;)
        (@FLAGS=PUBLIC;)
        (@RHS=
                (Write    ("c:\proto\hwymain.dbf")
        (@TYPE=DBF3;@FILL=NEW;@PROPS=Asphaltgrade_target,\
freeway_target,ID_target,Maintainer_target,\
Name_target,Roadcomp_target,Nolans_target,\
Speed_target,route1_target,Length_target;\
@FIELDS="Type","freeway","Mainhwy_ID","Admin1",\
"Hwyname","ComponentNo","No_lans","Max_speed",\
"route1","length";@ATOMS=<|Main_target|>;))
                (Write    ("c:\proto\hwysecd.dbf")
        (@TYPE=DBF3;@FILL=NEW;@PROPS=Asphaltgrade_target,\
freeway_target,ID_target,Maintainer_target,\
Name_target,Roadcomp_target,Nolans_target,\
Speed_target,route1_target,Length_target;\
@FIELDS="Type","freeway","secondaryhwy_ID",\
"Admin1","Hwyname","ComponentNo","No_lans",\
"Max_speed","route1","Length";@ATOMS=<|Secondary_target|>;))
                (Write    ("c:\proto\hwyothr.dbf")
      (@TYPE=DBF3;@FILL=NEW;@PROPS=Asphaltgrade_target,\
freeway_target,ID_target,Maintainer_target,\
Name_target,Roadcomp_target,Nolans_target,\
Speed_target,route1_target,Length_target;\
@FIELDS="Type","freeway","otherhwy_ID","Admin1",\
"Hwyname","ComponentNo","No_lans","Max_speed",\
"route1","Length";@ATOMS=<|Other_target|>;))
      )
)


(@RULE=   getobjects
      (@LHS=
                (=        (1)        (1))
      )
      (@HYPO=          loadreceivedobjects)
      (@RHS=
                (SendMessage     ("loadhighwayobjects_can")
      (@TO=|Highway_can|;))
      )
)

(@RULE=   getobjectstest
      (@LHS=
                (=        (1)        (1))
      )
      (@HYPO=          aggregate)
      (@RHS=
                (SendMessage     ("getrequestedobjects120_target")
      (@TO=|Highway_can|;))
                (SendMessage     ("getrequestedobjects100_target")
      (@TO=|Highway_can|;))
                (SendMessage     ("getrequestedobjects80_target")
      (@TO=|Highway_can|;))
                (SendMessage     ("setsimilarprops_target")
      (@TO=<|Main_target|>;))
                (SendMessage     ("setsimilarprops_target")
      (@TO=<|Secondary_target|>;))
```

```
            (SendMessage     ("setsimilarprops_target")
    (@TO=<|Other_target|>;))
            (SendMessage     ("writeobjects_target")
    (@TO=|Highway_target|;))
    )
)

(@RULE=   R_Send_query
    (@LHS=
            (Retrieve("c:\proto\form.nxp")        (@TYPE=NXP;))
            (=        (|Query_target|.Class2_target)         ("highway"))
    )
    (@HYPO=        Send_query)
    (@RHS=
            (Assign  ("highway")        (|Query_can|.class1_can))
            (Assign  (|Query_target|.Prop1_target)
    (|Query_can|.Prop1_can))
            (Assign  (|Query_target|.Prop2_target)
    (|Query_can|.Prop2_can))
            (Assign  (|Query_target|.Prop3_target)
    (|Query_can|.Prop3_can))
            (Assign  (|Query_target|.Prop4_target)
    (|Query_can|.Prop4_can))
            (Write   ("C:\proto\query.dbf")
    (@TYPE=DBF3;@FILL=NEW;@PROPS=class1_can,\
Prop1_can,Prop2_can,Prop3_can,Prop4_can;@FIELDS="class1_can",\
"prop1_can","prop2_can","prop3_can","prop4_can";\
@ATOMS=|Query_can|;))
    )
)
```

B.3 The Provider KB

The following is a list of the code of the provider knowledge base. It represents the export schema and context information of the database wich provides data to clients.

```
(@VERSION=        030)
(@PROPERTY=       Adminstrator_Source        @TYPE=String;)
(@PROPERTY=       Admn_class        @TYPE=String;)
(@PROPERTY=       class1_can        @TYPE=String;)
(@PROPERTY=       dum1    @TYPE=String;)
(@PROPERTY=       dum2    @TYPE=String;)
(@PROPERTY=       dum3    @TYPE=String;)
```

```
(@PROPERTY=        Fnode     @TYPE=Integer;)
(@PROPERTY=        Fnode_source      @TYPE=Integer;)
(@PROPERTY=        Length_Source     @TYPE=Float;)
(@PROPERTY=        Lpoly     @TYPE=Integer;)
(@PROPERTY=        Lpoly_source      @TYPE=Integer;)
(@PROPERTY=        Maintainer_Source          @TYPE=String;)
(@PROPERTY=        Name_Source       @TYPE=String;)
(@PROPERTY=        Nolan_Source      @TYPE=Integer;)
(@PROPERTY=        Prop1_can         @TYPE=String;)
(@PROPERTY=        Prop2_can         @TYPE=String;)
(@PROPERTY=        Prop3_can         @TYPE=String;)
(@PROPERTY=        Prop4_can         @TYPE=String;)
(@PROPERTY=        Roadid_source     @TYPE=Integer;)
(@PROPERTY=        roadname_source   @TYPE=String;)
(@PROPERTY=        Roadno_Source     @TYPE=String;)
(@PROPERTY=        roadparts_source  @TYPE=Integer;)
(@PROPERTY=        Roads     @TYPE=Integer;)
(@PROPERTY=        Roads_id          @TYPE=Integer;)
(@PROPERTY=        Route     @TYPE=String;)
(@PROPERTY=        Rpoly     @TYPE=Integer;)
(@PROPERTY=        Rpoly_source      @TYPE=Integer;)
(@PROPERTY=        Rte_num1          @TYPE=String;)
(@PROPERTY=        Rte_num2          @TYPE=String;)
(@PROPERTY=        Speedlimit_can    @TYPE=Integer;)
(@PROPERTY=        Surface_Source    @TYPE=String;)
(@PROPERTY=        Tnode     @TYPE=Integer;)
(@PROPERTY=        Tnode_source      @TYPE=Integer;)
(@PROPERTY=        Toll_rd   @TYPE=Boolean;)
(@PROPERTY=        Topspeed_Source @TYPE=Integer;)
(@PROPERTY=        Type      @TYPE=String;)


(@CLASS=  dumyclass
)

(@CLASS=  Gravel_Source
     (@PROPERTIES=
               Adminstrator_Source
               Fnode_source
               Length_Source
               Lpoly_source
               Name_Source
               Roadid_source
               Roadno_Source
roadparts_source
```

```
                Rpoly_source
                Tnode_source
        )
    )

    (@CLASS=  Interstate_Source
        (@SUBCLASSES=
                Pavedundiv_Source
                Gravel_Source
                Multilan_Source
        )
        (@PROPERTIES=
                Adminstrator_Source
                Fnode_source
                Length_Source
                Lpoly_source
                Name_Source
                Roadid_source
                Roadno_Source
                roadparts_source
                Rpoly_source
                Tnode_source
        )
    )

    (@CLASS=  Intrastate_Source
        (@SUBCLASSES=
                Paved_Source
                Multilan_Source
        )
        (@PROPERTIES=
                Adminstrator_Source
                Maintainer_Source
                Surface_Source
        )
    )

    (@CLASS=  Multilan_Source
        (@PROPERTIES=
                Adminstrator_Source
                Fnode_source
                Length_Source
                Lpoly_source
                Maintainer_Source
                Name_Source
                Roadid_source
                Roadno_Source
                roadparts_source
                Rpoly_source
                Speedlimit_can
                Surface_Source
                Tnode_source
                Topspeed_Source
        )
)

(@CLASS=  Pavedundiv_Source
        (@PROPERTIES=
                Adminstrator_Source
                Fnode_source
                Length_Source
                Lpoly_source
                Name_Source
                Roadid_source
                Roadno_Source
                roadparts_source
                Rpoly_source
                Tnode_source
        )
)

(@CLASS=  Paved_Source
        (@PROPERTIES=
                Adminstrator_Source
                Maintainer_Source
                Surface_Source
        )
)

(@CLASS=  USHighway_Source
        (@SUBCLASSES=
                Intrastate_Source
                Interstate_Source
        )
        (@PROPERTIES=
                Adminstrator_Source
        )
)
```

```
(@OBJECT=dumy
    (@PROPERTIES=
            dum1
            dum2
            dum3
    )
)

(@OBJECT=mapobjects
    (@PROPERTIES=
            Value    @TYPE=Boolean;
    )
)

(@OBJECT=replytoquery
    (@PROPERTIES=
            Value    @TYPE=Boolean;
    )
)

(@OBJECT=Start_Loading
    (@PROPERTIES=
            Value    @TYPE=Boolean;
    )
)

(@OBJECT=y
    (@PROPERTIES=
            Value    @TYPE=Date;
    )
)
(@METHOD=        sendobjectstocan_source
    (@ATOMID=Interstate_Source;@TYPE=CLASS;)
    (@ARG1=_classname1;@NATURE=Class;)
    (@ARG2=_classname2;@NATURE=Class;)
    (@ARG3=_classname3;@NATURE=Class;)
    (@FLAGS=PUBLIC;)
    (@RHS=
            (Execute ("AtomNameValue")
    (@WAIT=TRUE;@ATOMID=<_classname1>;@STRING="@RETUR
N=dumy.dum1,\
@NAMES";))
            (Execute ("LinkMultiValue")
    (@WAIT=TRUE;@ATOMID=dumy.dum1;@STRING="@LINKTO=h
ighway_can";))
```

```
                (Reset     (dumy.dum1))
                (Execute ("AtomNameValue")
     (@WAIT=TRUE;@ATOMID=<_classname2>;@STRING="@RETUR
N=dumy.dum2,\
@NAMES";))
                (Execute ("LinkMultiValue")
     (@WAIT=TRUE;@ATOMID=dumy.dum2;@STRING="@LINKTO=h
ighway_can";))
                (Reset     (dumy.dum2))
                (Execute ("AtomNameValue")
     (@WAIT=TRUE;@ATOMID=<_classname3>;@STRING="@RETUR
N=dumy.dum3,\
@NAMES";))
                (Execute ("LinkMultiValue")
     (@WAIT=TRUE;@ATOMID=dumy.dum3;@STRING="@LINKTO=h
ighway_can";))
                (Reset     (dumy.dum3))
     )
)
(@METHOD=          setexpressway_can
     (@ATOMID=Interstate_Source;@TYPE=CLASS;)
     (@FLAGS=PUBLIC;)
     (@LHS=
                (=         (SELF.Adminstrator_Source)          ("state
highway"))
     )
     (@RHS=
                (Assign  (FALSE)(SELF.Expressway_can))
     )
     (@EHS=
                (Assign  (TRUE)  (SELF.Expressway_can))
     )
)
(@METHOD=          setRoadType_can
     (@ATOMID=Interstate_Source;@TYPE=CLASS;)
     (@FLAGS=PUBLIC;)
     (@LHS=
                (Member(<|Highway_can|>)          (<|Pavedundiv_Source|>))
                (Member(<<|Highway_can|>>)        (<|Gravel_Source|>))
(Member         (<<<|Highway_can|>>>)    (<|Multilan_Source|>))
     )
     (@RHS=
                (Assign  ("paved undivided")
     (<|Highway_can|>.Roadtype_can))
                (Assign  (120)     (<|Highway_can|>.Speedlimit_can))
                (Assign  ("gravel")
        (<<|Highway_can|>>.Roadtype_can))
                (Assign  (80)     (<<|Highway_can|>>.Speedlimit_can))
                (Assign  ("multi-lane divided")
        (<<<|Highway_can|>>>.Roadtype_can))
                (Assign  (100)    (<<<|Highway_can|>>>.Speedlimit_can))
        )
)
(@METHOD=          setroute1_can1
        (@ATOMID=Interstate_Source;@TYPE=CLASS;)
        (@FLAGS=PUBLIC;)
        (@LHS=
                (=        (SELF.Adminstrator_Source)         ("state
highway"))
        )
        (@RHS=
                (Assign  (STRCAT("state hwy ",SELF.Roadno_Source))
        (SELF.Route1_can))
        )
)
(@METHOD=          setroute1_can2
        (@ATOMID=Interstate_Source;@TYPE=CLASS;)
        (@FLAGS=PUBLIC;)
        (@LHS=
                (=        (SELF.Adminstrator_Source)         ("interstate"))
        )
        (@RHS=
                (Assign  (STRCAT("interstate ",SELF.Roadno_Source))
        (SELF.Route1_can))
        )
)
(@METHOD=          setroute1_can3
        (@ATOMID=Interstate_Source;@TYPE=CLASS;)
        (@FLAGS=PUBLIC;)
        (@LHS=
                (=        (SELF.Adminstrator_Source)         ("US
highway"))
        )
        (@RHS=
                (Assign  (STRCAT("U.S. hwy  ",SELF.Roadno_Source))
        (SELF.Route1_can))
        )
)
```

```
(@RULE=   R_reply
    (@LHS=
            (=      (1)      (1))
    )
    (@HYPO=         replytoquery)
    (@RHS=
            (Retrieve ("c:\proto\query.dbf")
    (@TYPE=DBF3;@FILL=ADD;@PROPS=class1_can,\
Prop1_can,Prop2_can,Prop3_can,Prop4_can;@FIELDS="class1_can",\
"prop1_can","prop2_can","prop3_can","prop4_can";))
    )
)

(@RULE=   startmapping
    (@LHS=
            (=      (1)      (1))
    )
    (@HYPO=         mapobjects)
    (@RHS=
            (SendMessage    ("sendobjectstocan_source")
    (@TO=<|Interstate_Source|>;@ARG1=|Pavedundiv_Source|;\
@ARG2=|Gravel_Source|;@ARG3=|Multilan_Source|;))
            (SendMessage    ("setexpressway_can")
    (@TO=<|Interstate_Source|>;))
            (SendMessage    ("setroadtype_can")
    (@TO=<|Interstate_Source|>;))
            (SendMessage    ("setroute1_can1")
    (@TO=<|Interstate_Source|>;))
            (SendMessage    ("setroute1_can2")
    (@TO=<|Interstate_Source|>;))
            (SendMessage    ("setroute1_can3")
    (@TO=<|Interstate_Source|>;))
            (SendMessage    ("setsimilarprops_can")
    (@TO=<|Highway_can|>;))
    )
)

(@RULE=   startwriting
    (@LHS=
            (=      (<|Highway_can|>.Route1_can)
    (Query_can.Prop3_can))
    )
    (@HYPO=         writeobjects)
    (@RHS=
            (CreateObject    (<|Highway_can|>)         (|dumyclass|))
```

```
                (SendMessage     ("writeobject_can")
        (@TO=|dumyclass|;))
        )
)

(@RULE=    Start_Source
        (@LHS=
                (=        (1)        (1))
        )
        (@HYPO=          Start_Loading)
        (@RHS=
                (Retrieve ("c:\proto\ushwy.dbf")
        (@TYPE=DBF3;@FILL=ADD;@NAME="Route_'!Roads_!";\
@CREATE=|Gravel_Source|;@PROPS=Adminstrator_Source,\
Length_Source,Roadno_Source,Name_Source,Roadid_source,\
Fnode_source,Tnode_source,Lpoly_source,Rpoly_source,\
roadparts_source;@FIELDS="Admn_class","Length",\
"Rte_num1","Route","Roads_","fnode_","tnode_",\
"lpoly_","rpoly_","roads_id";@QUERY="type= \"gravel\"";))
                (Retrieve ("c:\proto\ushwy.dbf")
        (@TYPE=DBF3;@FILL=ADD;@UNKNOWN=TRUE;@NAME="Ro
ute'!Roads_!";\
@CREATE=|Pavedundiv_Source|;@PROPS=Adminstrator_Source,\
Length_Source,Roadno_Source,Name_Source,Fnode_source,\
Tnode_source,Lpoly_source,Rpoly_source,Roadid_source,\
roadparts_source;@FIELDS="Admn_class","Length",\
"Rte_num1","Route","fnode_","tnode_","lpoly_",\
"rpoly_","roads_","roads_id";@QUERY="type = \"paved undivided\"";))
                (Retrieve ("c:\proto\ushwy.dbf")
        (@TYPE=DBF3;@FILL=ADD;@UNKNOWN=TRUE;@NAME="Ro
ute'!Roads_!";\
@CREATE=|Multilan_Source|;@PROPS=Adminstrator_Source,\
Length_Source,Roadno_Source,Name_Source,Fnode_source,\
Tnode_source,Lpoly_source,Rpoly_source,Roadid_source,\
roadparts_source;@FIELDS="Admn_class","Length",\
"Rte_num1","Route","fnode_","tnode_","lpoly_",\
"rpoly_","roads_","roads_id";@QUERY="type = \"multi-lane divided\"";))
        )
)
```

CURRICULUM VITAE

Yaser Bishr, born on 15 August 1965, received his B.Sc. degree in Civil Engineering with specialization in Surveying Engineering in 1989, from Zagazig University, Faculty of Engineering, Cairo, Egypt. Few months later he became a lecturer in the same faculty where he was teaching geographic information systems, and geodesy. In 1991 he obtained his first post graduate diploma in computer programming and the theory of errors, from the same Faculty. During this period he was heavily involved, with a team from Finland, in a project for designing and developing the information system of the underground utilities in Cairo. From 1988 till 1991 he was teaching advanced programming, application of CAD systems in Civil Engineering, and database design at one of the largest computer centers in Egypt. In January 1992 he became a student at the International Institute for Aerospace Survey and Earth Sciences (ITC), Enschede, the Netherlands, where he graduated from the post graduate diploma in integrated mapping and geoinformation production. In December 1993 he obtained his M.Sc. degree, with distinction, in expert systems and spatial decision support systems and their application in watershed management from ITC. In 1994 he started his Ph.D. research project reported in this thesis. He is the author and co-author of several scientific articles which were published in International journals and conferences.

Propositions

Yaser Bishr, 1 October, 1997. Semantic Aspects of Interoperable GIS

(1) Information sharing, as a component of geographic information infrastructure, is a fundamental aspect in improving the efficiency and reliability of environmental decision making. It provides a framework for the development of multi-level decision support systems.

✍ ***This Thesis***

(2) Interoperability is not only among GIS software, but also among applications. The complexity of spatial information is one main factor which makes interoperability among applications so difficult to achieve.

(3) Spatial information is becoming a necessity in many aspects of our daily life, e.g., travel maps on the Internet, car navigation systems, etc. In the near future it will cover a wide spectrum if users from experts to novice. It is essential to develop software components which hide the complexity of the spatial information and provide an intuitive interface as well as representation.

(4) Interoperability is an array of levels, from network protocols to interoperability between communities. It has two main perspectives, the system perspective, and the data modeling perspective.

✍ ***This thesis***

(5) The key issue in GIS Interoperability is to resolve the semantic, schematic and syntactic heterogeneity. Resolving semantic heterogeneity is essential in order to resolve schematic heterogeneity. Resolving syntactic heterogeneity can be achieved by providing a unified geospatial data model, e.g., the formal data structure.

✍ ***This Thesis***

(6) The formalization of the context and ontology of each discipline, and the creation of a collection of proxy contexts abstracting them, provides a theoretical background for developing off-the-shelf semantic translators that can facilitate information sharing within and across disciplines.

✍ ***This Thesis***

(7) Semantic heterogeneity across disciplines can not be avoided. However, in the near future, the development of semantic translators, in the form of middleware, will implicitly create de facto standards for the sematnics per discipline.

(8) Knowledge is and will be produced in order to be sold, it is and will be consumed in order to be valorized in a new production: in both cases, the goal is exchange.

✍ ***Jean François Lyotard** (b. 1924), French philosopher*

(9) Information sharing pertains to answering three questions: "what" "where" and "how". Currently the power is the hands of those who can answer the "what" question, i.e., information providers. In the near future the power will be in the hands of those who can answer the "where" and the "how" questions, i.e., those who have the technology to comprehensively document, classify, and index the information of those providers.

(10) In the era of information sharing and interoperability, the known components of GIS which contain data input, data storage and retrieval, data manipulation and analysis, and data reporting, should be extended to include those components which facilitate information sharing and interoperability.

(11) There's only one corner of the universe you can be certain of improving, and that's your own self.

✍ ***Aldous Huxley** (b. 1894), British author*

(12) We did not inherit the Earth from our parents, rather we borrowed it from our children.

✍ ***Unknown***

(13) Information sharing is not only about cooperation nor only competition, rather it is about co-opetetion.