

Spatial Data Warehouse – A Prototype

Lionel Savary and Karine Zeitouni

Computer Science Department
Versailles Saint-Quentin University
PRiSM laboratory
45 avenue des Etats-Unis
78035 Versailles, France

{Lionel.Savary,Karine.Zeitouni}@prism.uvsq.fr

Abstract. Nowadays, there are an emergence of spatial or geographic data stored in several and heterogeneous databases, mostly in Geographic Information Systems (GIS). The diversity of GIS and the increasing accumulation of non-spatial (simple attributes) and spatial (geometric shapes) data make it difficult to apply conventional OLAP and data mining tools. Thus, the need to build a spatial data warehouse over heterogeneous GIS is becoming necessary in many fields. Our architecture is a central type architecture based on GML (for spatial data representation) and more generally on XML (for all data). In this paper we will focus on the study of data integration into a data warehouse, and data representation. We will show how the specificities of our architecture contribute to manage spatial and non-spatial data.

1 Introduction

Designing a data warehouse implies to face up problems like scattering information in different sources (Geographic Information Systems) and the ability of these sources to answer a particular query. One of the major problems encountered in the design of a data warehouse is the schema and data integration. In this case, the data warehouse designer must take into account the heterogeneity of data formats stored in a specific Geographic Information System (GIS). A common solution to overcome this problem is to convert all heterogeneous data into a single standard data model, once schema conflicts have been resolved.

This paper is organised in four sections. The second section enumerates the schema integration process and emphasises the problems encountered in schema and data integration for *non-spatial* and *spatial* data. In the third section, we will review and discuss about existing architectures and especially spatial data warehouse architectures. The fourth section details our prototype and highlights its specificity. Finally, we will conclude by giving the major contributions of our proposition.

2 Schema and Data Integration

The integration task refers to the problem of integrating the data from two or more different data sources. Integration is often required between applications or databases which differ in the way data are stored, the semantic of the data and the way it is organized. Two types of heterogeneity are distinguished: schema and data heterogeneity.

2.1 Schema Heterogeneity

Since database systems are generally developed independently, integrating different database schemas becomes difficult because of different structures, terminologies and focuses [2] used by each database designer. Moreover, in GIS case, managing heterogeneity among data is much more complex, because it must take into account *non-spatial* and *spatial* data [3]. Conflicts associated to spatial database include previous conflicts and specific conflicts due to [4]: different scales in spatial representations, different spatial referential, different geometry types depending on the specific point of view in a given GIS application.

2.2 Data Heterogeneity

Conflicts encountered at this level result usually from [1]: different data formats for the same field in semantic, abbreviation data encoding, missing values, duplicated information.

To face the problems of database and more specifically GIS interpretabilities, some methodologies have been defined and are commonly known as *integration process*. But *integration process* is difficult because conflicts at both structural and semantic level must be addressed. One can consider two integration levels: *schema* and *data integration*.

2.3 Schema and Data Integration

Before integrating data, one must first consider the *schema integration*. This phase consists in combining database schema into a coherent and global view. This operation is divided into four steps [1]: pre-integration, schema comparison, schema conformity, and finally schemas fusion and reorganization.

After this phase, *data integration* phase could start. Before the data being integrated into the data warehouse, they pass by the *preparation* phase. This phase is divided into three steps [1]: data extraction, data storage, and data cleaning. Data cleaning is one of the most important step in the *preparation* phase, and includes four principal functionalities: conversion and normalization of target data, field dependant cleaning, and rule based cleaning.

For integrating *spatial* and *non-spatial* data, specific components must be introduced into common data warehouse architecture, in order to manage these types

of data. In this field, only few works have been done and they only focus on managing *spatial* and *non-spatial* data.

3 Review of Existing Architectures

Existing approaches for managing *spatial* and *non-spatial* data are based on Geography Markup Language (GML) and more generally, in XML. The reasons of using XML as the common protocol for geo-referenced information exchange are as follows [7]:

- Text-based information exchange protocol ensures platform independence and easy implementation.
- XML may describe any data format of produced datasets or any GIS.
- The generated XML document is application independent, which can be used in any application as long as the application can parse XML documents.
- Storing and transferring Geo-Referenced Information (GRI) in XML could be more compact because usually there are many attributes in a whole dataset but few attributes are associated with each object.

Some works carried out by Gupta et al. [5] propose a three tier architecture composed of database wrappers, two levels of mediators and clients. The first level are composed of mediators appropriate to the specificities of the different sources (i.e. spatial mediator for GIS and another for image databases sources). The second level is composed of the main mediator which receives the client query, applies a set of rules to identify the spatial and non-spatial part of the query which are cut into sub-queries. These sub-queries are then routed to the specific mediator. The returned results are then gathered into a single document using the XMAS [6] language which allows the fusion of objects like pictures and maps into a new XML composite object.

Other works done by J.Zhang et al. [7], propose a prototype for wrapping and visualizing geo-referenced data in a distributed environment using XML technology. In this prototype, XML is used as a communication protocol between distributed web sites that provide GRI and the mediator, and between the mediator and clients. Java Servlets implement data translation into XML documents. Data in distributed websites can be stored in a flat file, relational database, object-oriented database or object relational database. A Java Servlet in the mediator server retrieves data from related distributed websites in an XML format upon a request from the client side, parses the retrieved XML documents, performs merge or other operations on the retrieved XML documents to build a new XML document and sends it to the client side. When the client side gets the requested data from the mediator server, it parses the returned XML document and draws it inside the browser window by using a Java applet.

In these two works [5], [7], the author suppose that spatial queries are processed by the sources and the wrappers, before being integrated in a common XML document with other data types. So, they do not implement any spatial query engine in the mediator itself. This limits the system by not allowing queries that combine different sources. To overcome this problem, we propose a new prototype also based on XML language offering more capabilities at the mediator level.

4 Our Prototype

Our prototype is a three tier architecture based on XML and GML. The communications between clients and mediator, mediator and wrappers are based on the Simple Object Access Protocol (SOAP). This protocol allows exchanging information over distributed environment. It is based on XML and can be generalized beyond HTTP. It can be implemented over CORBA/IIOP, COM, TCP/IP or SMTP and it is independent of any platform. SOAP can be used as a Remote Call Procedure (RPC) to send a message query and retrieve responses.

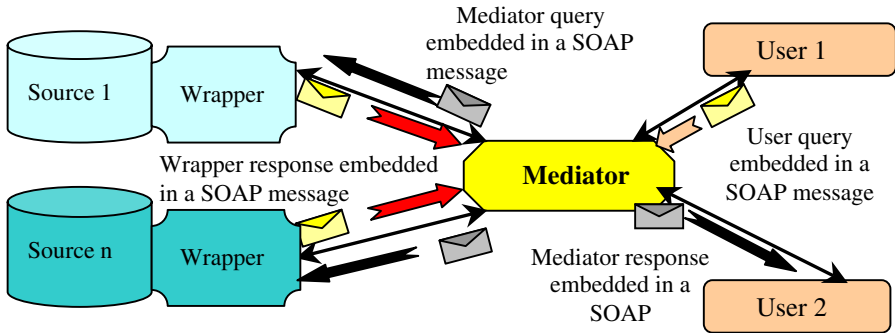


Fig. 1. Three tier architecture based on SOAP protocol

The three main components of the architecture in figure 1 are composed of sources and wrappers (the first tier), mediator and the mediator database (the second tier), users (the third tier).

4.1 Source and Wrapper Level

At this level, each data source is composed of a GIS like ArcView or a database like Oracle and a wrapper. The wrapper transforms data stored in an original format (such as ArcView or Oracle formats) into XML/GML format. Because ArcView data are stored in flat files, they could be accessed in any programming language without on-line access to the GIS. Whereas in Oracle, a connection to the database is required. *Non spatial* data are coded into XML format and *spatial* data into GML. These two types of data are embedded in a SOAP message and are sent to the mediator. If the capabilities of the sources are limited, for queries that can not be processed by the source, the wrapper should provide all capabilities of XQuery and spatial queries. It returns all the data in XML/GML format to the mediator. Otherwise, if the source allows handling XML data, but not spatial queries and operators, the non-spatial part of the query could be easily wrapped in the specific language of the source (such as `DBMS_XMLQuery` for Oracle), whereas the spatial part necessitates the implementation of spatial engine within the wrapper, generating a GML result.

4.2 Mediator Level

4.2.1 Mediator Components

The purpose of our work is to propose an architecture based on standard components that allow to plug in any other component without any modification of the whole or a part of the architecture. To achieve this goal, the mediator is based on standard components like Java Topology Suite (JTS) and XQuery. JTS [8] is an open source conform to the Open GIS Consortium features specifications. It implements spatial predicates and functions for managing two dimensional data stored in GML format. Thus, we are using JTS to deal with *spatial* data, and XQuery for *non-spatial* data. The advantage of these components is the allow the support any type of databases at this level since all operations are computed by specific components. Oracle has been chosen for the mediator database, for the moment, because it holds the management of XML database (XML_DB) and should optimise data retrieval using specific indexes. The figure below summarises this prototype architecture.

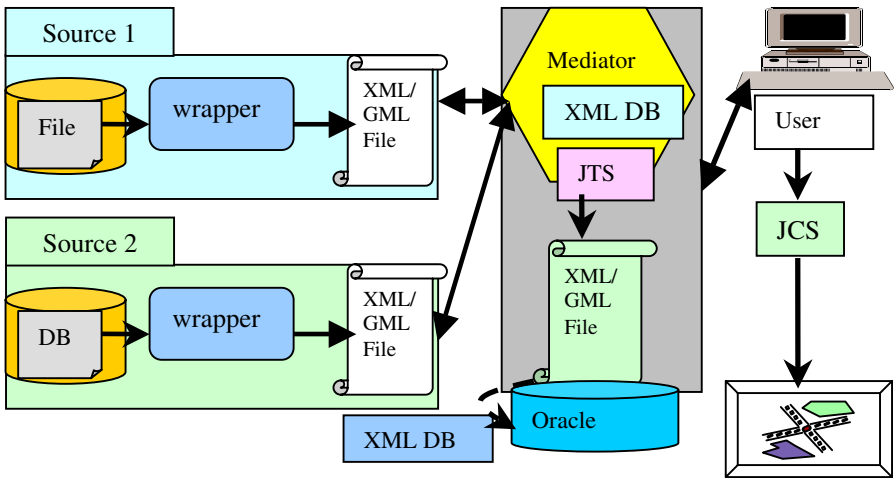


Fig. 2. Prototype architecture

4.2.2 Query Processing

Once the SOAP message containing one user query is received by the mediator, the XML documents is extracted. Then the mediator first queries the Oracle database. If this mediator database can not answer the user query, the query is then split into sub-queries and sent to the distributed servers concerned by these sub-queries. The returned results from the distributed servers are then processed by the mediator. If no more treatment is required, the results are gathered into a single XML/GML document, stored in the XML_DB and rooted to the user. If any treatment is required, the mediator computes the required operations before merging the result into a single XML/GML document.

4.3 Client Level

The message received by the user is then extracted and displayed using Java Conflation Suite (JCS). JCS is an open source conform to the Open GIS Consortium features specifications. It allows a graphical visualization of geographic data and supports GML and operations on graphical display such as zoom-in, zoom-out, etc.

5 Conclusion

This article has pointed out the problems encountered during *non-spatial* and *spatial* data integration into a data warehouse. *Spatial* data are more complex and more difficult to be integrated and require specific architecture. In the review section, it has shown existing architectures for *non-spatial* and some for *spatial* data. But these XML-based architectures only support *spatial* data management at the source or wrapper level. One solution could be to process the whole query at the level of the warehouse database (for example Oracle), and convert the result into GML format. However, this solution is not optimal because of the data flow volume. Our architecture overcomes this problem by using XML SQL Utility, Java Topology Suite, Java Conflation Suite and SOAP. Using these technologies, the mediator capabilities and performances are enhanced in answering a *spatial* query and the client allow the users a powerful graphic interface and map viewing tool.

References

1. Fundamentals of data warehouses, Springer edition, 2000
2. P.Johannesson, M.Jamil.: Semantic interoperability context, issues, and research directions. In: Second International Conference on Cooperating Information System, Eds. , 192–199. 1984
3. schema integration methodology and Toolkit for Heterogeneous and Distributed Geographic Databases. In: Journal of the Korea Industrial Information Systems Society, V6, 3 (September 2001), 51–64
4. *Data* and Metadata: Two-Dimensional Integration of Heterogeneous Spatial Databases .In: Spatial Data Handling 98 Conference Proceedings, Vancouver, BC, Canada, July 1998, 172–179
5. A.Gupta, R.Marciano, J.Zaslaviska, G.Baru: Integrating GIS and Imagery through XML-based Information Mediation. In: Digital Images and GIS, Lectures Notes In Computer Science, 1999, Vol.1787
6. C.Baru, A.Gupta, B.Ludäscher, R.Marciano, Y.Papakonstantinou, P.Velikhov, A.Yannakopoulos: XML-based Information Mediation with MIX. In: proceeding of the SIGMOD, 1999
7. J.Zhang, M.Javed, A.Shaheen, L.Gruenwald: Prototype for Wrapping and Visualizing Geo-Referenced Data in a Distributed Environment Using XML Technology. In: Eighth International Symposium of ACMOS, McLean, Virginia, November 2000
8. <http://www.vividsolutions.com/jts/jtshome.htm>