Management Information Systems, C.A. Brebbia & P. Pascolo (Editors)

Object oriented database for a GIS

N. Posada & D. Sol
CENTIA, Universidad de las Américas, Puebla, Mexico

Abstract

This paper describes the design and implementation of geographic data in an object/relational database. Infomix Universal Server (IUS) is a tool that permits the description of persistent geographic objects. The geographic objects were designed by using the OMT methodology. The application was built with the Java programming language. The objects were made persistent by using the IUS database management system. A web user interface was developed to show the geographic data. Data modeling was done by using points, lines, polylines and polygons. The data stored in the database could be exported and used by a client who runs a web user interface. Our representation gives the description to have an homogeneous geographic data representation to be used in a geographic database server. The user can then access and use the data stored in the database.The web user interface runs on the client and the database manager runs on the server. The object oriented geographic databases that use a standard representation are the first step to build tools that will permit the interoperability of heterogeneous GIS.

1 Introduction

GIS (Geographic Information Systems) is the term to designate the systems that work with spatial and geographical data (15). The real world objects can be represented by using several geometries. With the use of new geometries we will have more accurate mechanisms to represent geographic data (10).

The databases are used normally to store different kind of data. The modeled domain determines the kind of data that is represented. The models used today in a database assure consistensy, security, reduce redundancy and permit the concurrent access. The data stored in a database is different from the data used in a real world. Mechanisms to store and to retrieve data must be developed.

The development of the object oriented paradigm programming suggests the use of the object oriented approach to be considered in a database. The object oriented DBMS are created to support CAD applications, multimedia systems like GIS and to handle voice, video and normal data. This paper describes the use and management of geographic objects in a database. The DBMS used is the object/relational Informix module. A web user interface developed in our research group (3) presents the geographic data. The interface manages several GIS aspects like pan, geometric operations (translation, zoom), scales and coordinates. Our work can be a platform to test other aspects like queries, interoperability and sharing of geographic data.

2 Related work

The research developed in this area shows specific and general characteristics. An important aspect is the fact that interoperability needs the use of a standard format for geographic data. The general characteristics show that there is an interest to establish a standard. We try to indentify the important aspects to be included in our work. Table 1 shows the works that have been analyzed with their principals properties.

The principal characteristics are: aggregation, association, generalization, classification and inheritance. In particular (5) shows how the inheitance and the association are modeled.

Model	Classification	Generalizaition	Association	Aggregation	Inheritance	Propagation	Display
Object-Oriented Modeling for GIS (5)	■	■	■	■	■	■	
Rock & Roll (6)			■				
Computer Cartography for GIS: an Object-Oriented view on the Display Transformation (1)	■				■		■
Object Oriented Modeling for Geograph Information System (OOGIS) (12)	■	■	■	■	■		■

Table 1 Features of work analyzed and system OOGIS proposed

3 Methodology

We used the OMT methodology (13) because it has the facility to manage objects in the design step. These objects can be used in programming and databases in the object oriented approach. Our goal is to handle geographic objects in three levels: design, programming and databases. At the same time we keep the advantages offered by the relational model.

4 Data Model

Two approaches were considered to implement our ideas:

- object oriented database systems
- object/relational database systems

First we will describe some aspects of the relational approach that permit to understand easily the advantages of the object/relational approach.

4.1 Relational model

The relational model is very popular because this is a very simple approach. It has only one structure: a table with tuples and attributes with atomic data types. The query language uses simple operations on the tables and the more complex like joins don't need be understood by users (4). If the relational model is used the data needs to be atomized in rows and columns. The mechanisms to store and to retrieve data in the databases needs to be developed. To model one or two geographic objects the representation can use 4 or 5 relational tables. A GIS handles its data by layers. A map can be composed by rivers, roads and cities. Each kind of geographic object is represented by a layer. In this context to model several layers means that an important number of tables need to be implemented. For instance, a polygon is modeled by three or more segments and each segment is modeled by two points. The tables to model polygons, segments and points should be created. This kind of representation reduces the data clarity. It is possible to guarantee integrity and consistency but at the expense of clarity. The object oriented approach propose an option that tries to keep integrity and consistency but also clarity in the representation.

4.2 Object oriented model

This approach was inspired from the object oriented programming languages. The principal object oriented languages are Smalltalk, C++ and Java (2). The object oriented paradigm includes the abstract data type concept. The abstract data type declarations are public or private. In an object oriented language the types are implemented by classes. The object operations can also be public or private. An object is a black box, that is built and modified independently of other objects. As the object oriented paradigm is developed new approaches are proposed to implement the concepts related to the object paradigm. Each approach can manage several object concepts in a different way (8). An example are the tools offered by an object oriented language to implement encapsulation. The approach used to model the objects could have an influence on the object management in the database.

4.3 Object/Relational model

The more direct way to implement objects in a database and to keep the DBMS advantages is the Object/Relational model (14). A standard for a multimedia data manager is developed in the context of the SQL standard (11). To manage geographic data in an object database, the data types are first created. A hierarchy is used to represent from the basic data types to the more complex. All

the data types are then implemented by classes in an object oriented language programming. In the context of an object database the data that reference the data types is stored in tables, but the structure of the data is represented by the types.

5 Geographic objects

The basic geographic objects that we use are:

- points
- lines
- polylines
- polygons

Every kind of object is modeled by an abstract data type. We use the vectorial geometry to represent the geographic objects. This kind of modeling matchs the representation used by the user interface. The geometry used in the interface and in the database is based on standard geographic data. The database and the interface shall be used in the context of an interoperability GIS. Figure 1 shows how the basic geographic objects are used to model the Universidad de las Américas campus.

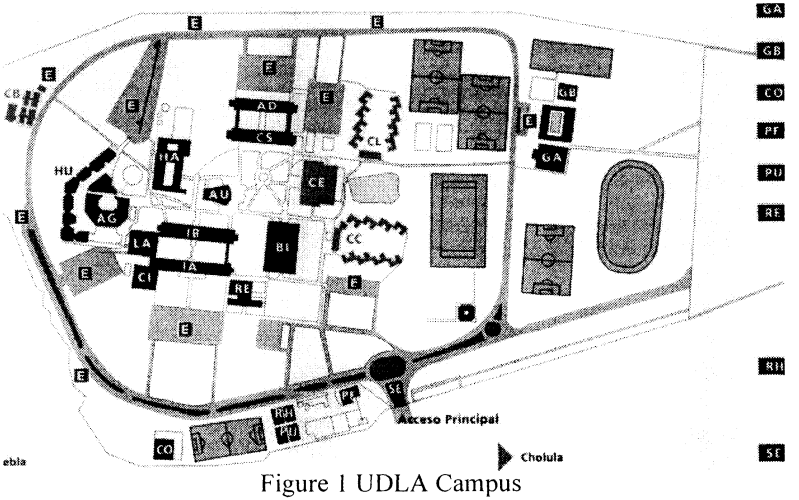

Figure 1 UDLA Campus

6 OOGIS model

The principal contribution of our work is the implementation of persistence for geographic objects. Figure 2 shows the class hierarchy. It shows in the first level the View class. Each View class is formed by several layers. In the second level we then represent the Layer class. Each Layer class can be modeled by one and only one data type. A brief description of the geographic objects is the following:

Management Information Systems, C.A. Brebbia & P. Pascolo (Editors)

- **Point:** It is composed by two coordinates X and Y.
- **Line:** A line is composed by two points, it means two pairs of X,Y coordinates. The aggreation of two points permits the creation of one line. From the Line class other more specific classes can be derived. These classes manage more attributes, for instance a straight road can be modeled by a line. The road name, the road number and the county are examples of descriptive data.
- **Polyline:** A polyline is composed by two or more lines. A polyline is an aggregation of lines. Like a line, a polyline can derive more specific classes. The river class is an example of an object that is a polyline. Descriptive attributes are used to model additional information.
- **Polygon:** A polygon is similar to a polyline but it is closed. The basic geographic data polygone doesn´t give a description for a map. More specific types (derived from polygon), like lakes, states or other object that can be represented by a polygon, will give descriptive map information.
- **Layer:** A layer is composed by the aggregation of two or more objects of the same type. For instance, the lake layer, the road layer and the county layer.

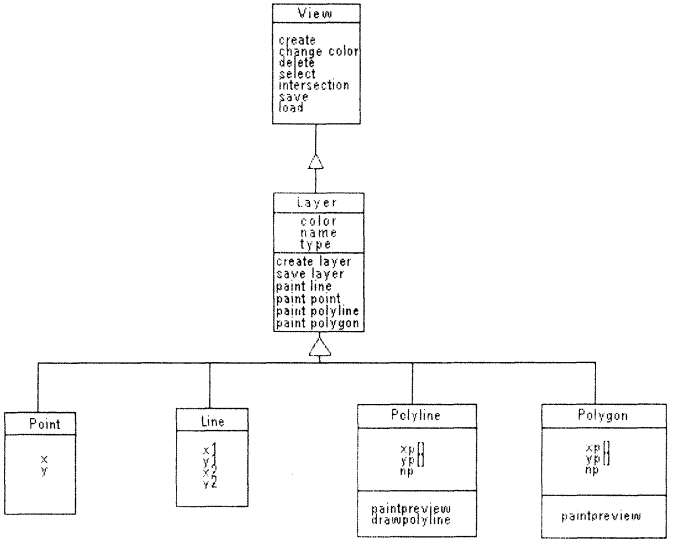

Figure 2 OOGIS Class Hierarchy, OMT design (4)

- **View:** A view is composed by the aggregation of two or more layers. The view class manages the activation of layers for edition. In this case it is possible to add new geographic objects. In our application, this class manages a method to do the intersection of two classes. The next section describes how all the objects are organized in the database.

Management Information Systems, C.A. Brebbia & P. Pascolo (Editors)

7 Database construction

The database is composed by two parts, the data type description and the tables. The tables make reference to data types to describe their structure. The data type description shows their attributes. The objects are stored in the tables, but they make reference to the corresponding data types for their description.

Name	Type	Table	Variable	Inheritance	Association
point_t	Yes	Yes	x, y		
layer_t	Yes	Yes	name, color, type		
layerpoint_t	Yes	Yes	x, y	layer_t	
layerline_t	Yes	Yes	end1, end2	layer_t	point_t
layerpolyline_t	Yes	Yes	num, end1, end2	layer_t	point_t
view		Yes	name_view, name_layer		
layer		Yes	name_layer, type		view

Table 2 OOGIS Types and Table used

The principal types used by our application are point and line. The types polyline and polygon are formed by aggregations of the basic types. More complex types are formed by inheritance and association. Table 2 shows the data type name, the attributes and the relationships with the other data types. The tables of the database will be created by reference on the data types. Two basic tables were created: View and Layer. These tables are built independently of any type. The View table manages the name of the layers to describe a map. The Layer table is responsible to manage its type. This table provides the access for any layer on the map.

8 Prototype

Our model was tested by using the Java object oriented language program and IUS DBMS (7) on Solaris. Our application has the following modules:

- the operations manager
- the database server
- the interface manager
- the modeling manager

Our prototype was tested with data from the Universidad de las Americas, campus. The prototype architecture is shown in figure 3. An example of the creation of tables and the data insertion is as follows:

```
create table Circuit of type layerpolyline_t:
insert into Circuit values ('Circuit','java.awt.Color[r=0,g=0,b=0]','polygon',0,
row(108,93)::point_t, row(80,95)::point_t):
```

```
create table Building of type layerpolyline_t;
insert into Building values
('Building','java.awt.Color[r=0,g=0,b=255]','polygon',0,row(58,69)::point_t,
row(79,70)::point_t);
```

The spatial data are stored in the database in three steps:

1. The layer names are associated to one project name. The table that implements the view type store the layer and project name association. This operation is done by the save method which belongs to the view class.
2. The view class save method calls the layer class save method. This method first stores in the layer table (which implements the layer class) the type and the name of the layer.
3. The same method (layer class save method) creates the table to store the layer attributs like color, name and description data type. This table implements the different geographic object data types (point, line, polyline, polygon). For each layer the method generates one table.

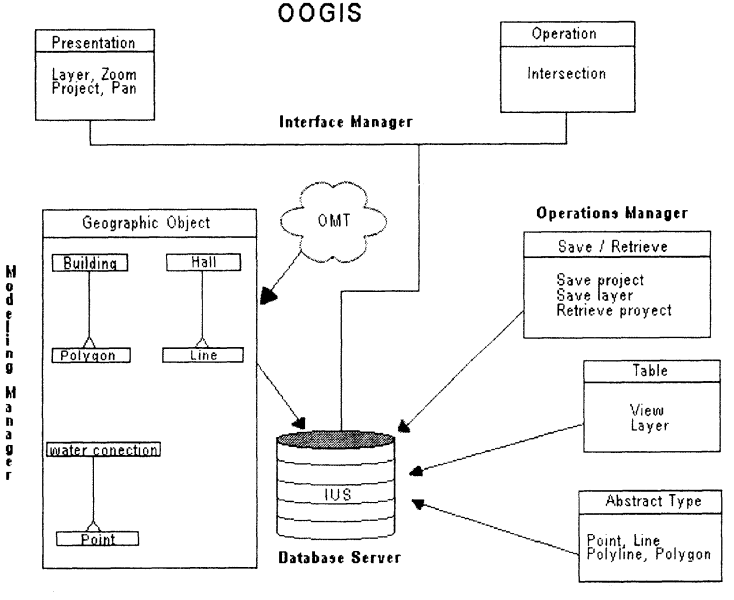

Figure 3 OOGIS Architecture

The types created in the database match the classes implemented in Java. It is possible to see that the types, the classes and the tables have a homogeneous description. The types described in the database are the same that the types described by the OMT design. With the use of a standard for geographic data like OpenGis types our approach will be tested widely. In particular interoperability and data exchange can be improved by this kind of representation.

The user interface is shown in the figure 4. It is possible to see the tools to manage zoom, pan and the geometric operations. On the right it is possible to control the layer activation for quering and edition. The application manages only a property format (points, lines, polylines and polygons). Actually it is modified to manage the geographic data in the OpenGIS format (9). A module to translate from shp ArcView is actually developed and very soon the applications will be capable to manage (in the database and in the interface) geographic data imported from ArcView. The application will be then capable to sopport several database users.

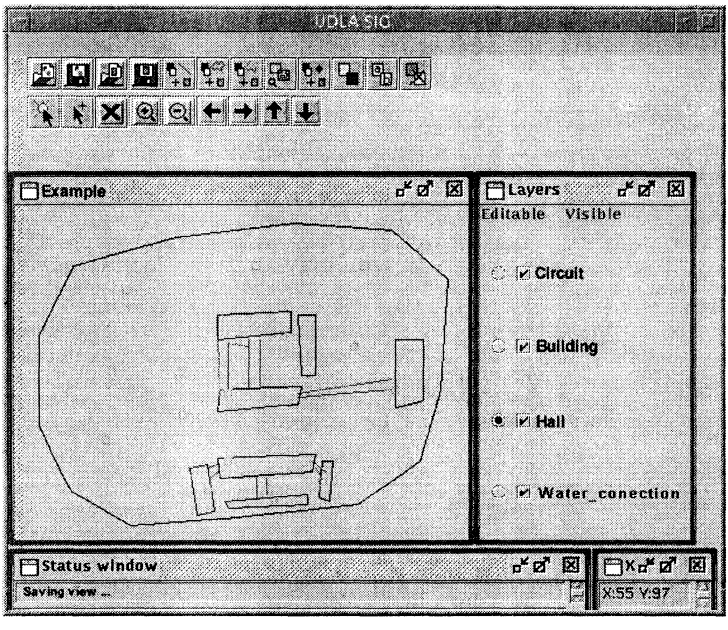

Figure 4 OOGIS Interface

9 Conclusion

Our principal goal is a transparent management of geographic objects in a database. The geographic objects can then be persitent objects. The objects don´t need to be stored and retrieved by the use of specific mechanisms. The data types created in the context of the database are referenced by the tables. The data types are implemented by classes in the context of the application. The previous work implemented geographic objects in applications but they don´t manage the persistence. The number of access to the database can be reduced. We think that the clarity of the representation is also improved. The objects represented in the database are objects that can be exported and shared with other applications. In particular, the use of a standard representation permits this interoperability. Actually our group works in the implementation of geographic objects in our database modeled in the OpenGIS standard representation. Heterogeneous applications could then share geographic objects.

Our application at this stage is a prototype. The performance seems good. For not complex geometric objects the data can be stored and retrieved in seconds. For complex objects in the case of remote connection our prototype depends on the network performance.

References

[1] Andrew U. Frank, Max J. Egenhofer, *Computer Cartography for GIS: an Object-Oriented view on the Display Transformation,* Computer & Geociences Vol 18, No.8 pp 975-987, 1992.

[2] Berard, Edward V. *Basic Object-Oriented Concepts,* http://www.toa.com/pub/html/oobasics/oobasics.html, 1996.

[3] Briones del Río, Juan Luis, *Interfaz gráfica para un Sistemas de Información Geográfico,* Computer Engineer thesis, UDLA, December 1998.

[4] Cattell, Roderic Geoffrey Galton, *Object Data Management: Object-Oriented and Extended Relational Database Systems,* Addison-Wesley, United States of America, 1991.

[5] Egenhofer Max J., Frank Andrew U., *Object-Oriented Modeling for GIS,* URISA Journal, 1998.

[6] Fernandes, Alvaro A. A., Paton, W. Norman, Howard Williams M., *A Logical Query Language for an Object-Oriented Data Model,* Department of Computing and Electrical Engineering, Heriot-Watt University, Edinburgh, UK, January 1999.

[7] Informix Software, Inc. *Answer on line*, http://ict2.udlap.mx/informix/, 1997.

[8] Khoshafian Setray, Abnous Razmik, *Object Orientation*, Second Edition, Jhon Wiley & Suns, Inc., United States of America, 1995.

[9] Kottman, Cliff, *OpenGis*, Open Gis Consortium, Inc., http://www.opengis.org, 1995.

[10] López Ornelas, Erick de Jesús, *Modelación de Información Espacial y Geográfica*, Computer Engineer thesis, UDLA, May 1998.

[11] Manola, Frank, Sutherland, Jeff, *SQL3 Object Model*, http://www.objs.com/x3h7/sql3.html, 1997.

[12] Posada Toledo, Nidia, *Modelado de Datos Orientado a Objeto para un Sistema de Información Geográfica*, Computer Engineer thesis, UDLA; May 1999.

[13] Rumbaugh James, Blaha Michael, Premerlani William, Eddy Frededick, Lorensen William, *Modelado y diseño orientados a objetos*, Prentice Hall, Spain, 1996.

[14] Stonebraker, Michael, Paul Brown, *Object Relation DBMS*, Ed. Morgan-Kaufmann, 1999.

[15] Vélez Macías, Fabio., *Introduction to GIS,* University of Antioquia, Department of Enviroment and Health Engineer, http://quimbaya.udea.edu.co/~fabiovel/, Medellín, September 1999.

Management Information Systems, C.A. Brebbia & P. Pascolo (Editors)

Section 8
Remote Sensing