An Implementation Approach for Object-oriented Topographic Databases using Standard Tools

Babak Ameri Shahrabi[1], Wolfgang Kainz

Department of Geoinformatics
International Institute for Aerospace Survey and Earth Sciences ITC
P.O. Box 6, 7500 AA Enschede, The Netherlands
phone: +31 53 874-434, fax: +31 53 874-335
e-mail: ameri@itc.nl & kainz@itc.nl

Abstract

The creation of (national) topographic databases is one of the high priority tasks of mapping agencies. There are already numerous projects in many countries such as the United States of America, France and Germany. They all use different approaches, but mainly based on conventional data models and database management systems.

This paper focuses on the design of an object oriented data model for large topographic databases and its implementation by using conventional database and GIS technology. The proposed approach is tested in the context of the national topographic database in the scale of 1 : 25000 of Iran. It is part of a larger project looking into alternative ways of implementing the national topographic database by using different approaches for data modelling and database querying.

Introduction

Manually drafted maps were the primary media to represent the location and relationships of geographic objects. As the user's knowledge of map capabilities increases analog maps proved to be insufficient for fast and effective extraction of information for geographic knowledge. Consequently, it became necessary to convert the knowledge stored on paper maps to digital form.

Within the realm of geographic data handling, this trend (increasing reliance on the computer as a data handling and data analysis tool) has been driven by both push and pull factors. The primary one was the push away from the limitation of manual techniques and the pull toward the use of computers (Peuquet and Marble 1990). The substantial improvement in computer systems during the last two decades has made it much easier to apply computer technology to the problem of storing, manipulating and analyzing large volumes of spatial data. Today many national survey organization make routine use of what are called topographic databases to undertake tasks such as production of base map series, more efficient and cost effective map updating, and support several user needs which deal with the analysis of spatial information. In general terms, a topographic database contains an image or model of real world phenomena. This model is an organized collection of related data about geographical

[1]Permanent address: National Cartographic Center of Iran (NCC), P.O.Box 13185-1684, Tehran, Iran. Fax: +98 21 6001971

features which can be divided in two types, spatial (geometry) and non-spatial (alphanumeric attributes) data. Accordingly, two types of database models should be distinguished: hybrid and integrated models (Healey 1991, Wessels 1993).

Hybrid systems store spatial data in a set of independently defined operating system files for direct, high speed access. Each file (also called maplayer or maptile) represents a selected set of closely associated topographic features (e.g. river, railway, boundary). Non-spatial data are stored in a standard database, usually a relational one which is linked to the spatial data through identifiers.

Integrated systems store both the spatial and non-spatial data in a (relational) database. These systems do not define independent layers or tiles, but just one seamless map. Relational tables hold the coordinates, topological information and attributes.

The efficient management of very large geographic and topographic data sets is a highly complex task that current commercial systems have great difficulties with. Conventional DBMSs usually support data models that are not very well suited for the management of geographic information. Geographic objects are often structured in a hierarchical manner. For example a province consists of several cities which are split up into districts, which in turn consist of streets, blocks, etc. Neither conventional DBMS can provide sufficient support for these complex geo-objects. Instead the user typically has to split up objects into components until they are atomic. This fragmentation considerably complicates the modeling which consequently complicates the interpretation of modeled objects and may have negative effects upon the performance of the system.

Recent research in software engineering has promoted an object-oriented design method by which real world objects and their relevant operations are modeled in a program which is more flexible and better suited to describe the complex real world situation. Object-orientation is a particular view of the world which attempts to model reality as closely as possible to applications (Webster 1990, Wessels 1993). The notion of an object made its first appearance in the early 1970s as the part of the simulation language SIMULA. But, since its influence is detectable in several distinct branches, it is also true that developments within those disciplines have helped to refine the set of unifying concepts that define the object-oriented approach and it should no longer be seen solely as a programming style (Webster 1990).

Object-orientation is applied to information technology in several ways (Kim and Lochovsky 1989, Bhalla 1991). Possible applications, relevant for spatial information systems are object-oriented programming languages, object-oriented databases and user interfaces (Worboys et al. 1990, Kemppainen and Sarjakoski 1990, Khoshafian and Abnous 1990).

Tools

At the present time most geographic information system designers apply relational database technology to implement their model for two important reasons. Firstly, relational database management systems are dominant on the database market and enjoy wide acceptance among database users; they are a 'de facto' standard for data processing applications (Wessels 1993). Secondly, internationally standardized tools, such as SQL, have been established for relational DBMSs which provide features for defining the structue of the data, modifying the data, and specifying security constraints within the database. Users can retrieve data by specifying in SQL what data are required. The system will work out the way how the requested data is retrieved (Date 1986).

However, object-oriented DBMSs are a new database technology and are therefore not much present on the database market. The fact that standards, as those for relational DBMSs are still not defined, and also as most of the organizations already have an information system based on a relational DBMS technology, the former approach is still the most important technology applied to spatial information systems. Whilst there is some general agreement on the characteristics of object-oriented databases, there is no consensus on how an object-oriented database should be implemented (Worboys et al. 1990). At present there are no full-scale implementations, but a number of research efforts are in progress of creating object-oriented databases which can be classified into three categories (Bhalla 1991):

(1) Those that are directly based on the object-oriented paradigm like ONTOS (Ontos 1991),
(2) extensions to relational systems such as POSTGRES (Stonebraker et al. 1990), and
(3) experimental toolkits (or storage system) such as EXODUS (Carey et al. 1986).

The second category indicates that relational DBMSs have developed a variety of techniques to facilitate object management within a database environment. These include storage of the structure of object hierarchies in relational tables, inheritance of instance variables and methods, storage of query language or programming language procedures representing instance methods, as specified fields in relational tables, and support for abstract user defined data types and operations supported on them. It should be considered that this approach is an evolutionary method as the best way forward to add object-oriented facilities to an existing relational database framework. This is in contrast to some of the proponents of object-oriented programming, who wish to achieve the same kind of functionality, but without the perceived constraints of the relational model (Healey 1991).

Query languages are very important for the retrieval of data to satisfy certain immediate user needs which were not planned for and for which no programmed access procedures are available. Interactive languages for ad hoc queries are usually included in conventional DBMSs. The SQL (Structured Query Language) whose original version was known as SEQUEL is the language which rapidly emerged as the standard database language and is widely implemented. SQL allows users to work with higher level data structures. Rather than manipulating single rows, a set of rows can be managed. SQL commands accept sets or rows as input and return sets as output. This set property of SQL allows the results of one SQL statement to be used as input to another. SQL does not require users to specify the access methods to the data. This feature makes it easier to concentrate on obtaining the desired results. A number of attempts have been made to extend SQL to make it useful for spatial databases (Egenhofer and Frank 1989, Egenhofer 1991). The PL/SQL, a procedural language extension to SQL (ORACLE 1992), fully integrates modern software engineering features which are practical tools for system developers and designers to bridge the gap between conventional database technology and procedural programming languages.

Another aspect is the programmers interface or procedural programming available in most GISs to provide a tool for developing user-specific interfaces such as AML (Arc Macro Language of Arc/Info), or a graphical user interfaces such as ESRI's ArcView.

Implementation

The object-oriented topographic data model is based on the object concept. The model represents a real world phenomenon of whatever complexity and structure as an object (Dittrich and Dayal 1986, Bonfatti et al. 1993). An object is characterized by encapsulating its structural/static properties (attributes, relationships, components) and its behavioral/dynamic properties (rules, operators, procedures) (Egenhofer and Frank 1988, Webster 1990).

Data Modelling

The components of design of an object-oriented model are as follows :

I- Object identification
Objects are the individual elements in the model which are defined for applications. In this research the objects are based on the specification and the contents of the base map series 1:25000 of Iran as a minimum requirement. However, they are modified in such a way to satisfy as much as possible other user groups.

II- Classification
Classification is the mapping of several objects into a common class. All objects that belong to the same class are described by the same properties and undergo the same operations. For example, in the transportation network, all highways which have the same properties (structural and behavioral) can be classified as a class *highway*.

III- Generalization and Inheritance
Similar to the concepts of object and class, the object-oriented model also supplies the dual concepts of generalization and specialization. Together, these two allow classes to be arranged in a hierarchy. A class which is a descendant of another class in the hierarchy is said to be a more specialized subclass, while the more general class is the superclass. A subclass has automatically the characteristics of the superclass defined for it but it may have also specialised characteristics of its own. Subclass and superclass are related by an IS_A relationship. For example, the classes *highway*, *main road*, *secondary road* could be grouped as a superclass *road*. In generalization hierarchies, some of the properties and operations of the subclasses depend on the structures and properties of the superclass(es). Properties and operations which are common for superclass and subclasses are defined only once, and the properties and operations of the superclass are inherited by all the objects of the corresponding subclasses. This concept is very powerful, because it reduces redundancy and maintains integrity. The inheritance can be single or multiple. Single inheritance requires that each class has only one direct superclass. In case of multiple inheritance, one subclass has more than a single direct superclass. For example, an object channel could be a subclass of two different superclasses, such as *water stream*, and *structure*. It will inherit properties from both superclasses. Sometimes the same property operation can be defined differently in two relevant superclasses, which is known as an inherit conflict. For

solving these conflicts, some special conditions must hold or a priority rule must be defined.

IV- Aggregation
In addition to elementary objects composite objects may exist. They can be defined through aggregation. An aggregation shows how composite objects can be built from elementary objects and how these composite objects can be put together to built more complex objects and so on. Suppose that we have houses, roads, parks as simple objects, then from these we can build complex object residential districts. The fact that the simple objects can be aggregated into complex objects implies that also their attribute values may be aggregated. If one of the attributes of object house is the number of people living there, then it is easy to calculate the total number of inhabitants of a residential district.

V- Association
Association or grouping is the construct which enables a set of objects of the same type to form an object of higher type. Association is used to describe the logical relationship between objects (classes). The association can be established between two or more classes or recursively in one class. For example, the association between road and railway is an association which describes that, road and railway are crossing each other. A road might cross the same railway several times. Both road and railway are members of the relevant association.

The features of interest in the topographic model include 9 classes at the top level. These spatial information classes are :

Survey control point consists of information about the points of established position that are used as fixed references in positioning the features. This class of objects includes three subclasses such as *geodetic control point* (planimetry), *bench marks* (altimetry), and the object class *full control point* (both planimetry and altimetry).

Vegetation consists of information about vegetative surface cover. It will be specialized to classes such as *forest, orchard, arable land* and so on. Vegetative features associated with wetland such as marsh and swamps are collected under class *hydrography*.

Building consists of information about the object building which means any permanent walled and roofed construction. This is a class including several subclasses such as class *educational building*, *medical building* etc. that each one is again a generalization of several classes. For example the object class *hospital*, or *clinic* are subclasses of class *medical building*.

Hydrography describes all information about the hydrographic features and includes *water course*, *water bodies*, and *water surface points* as its subclasses. These classes again are specialized to several classes. For example the *water course* is a superclass of object class *river*, *canal* etc.

Delimiter consists of all boundaries and delimiters including virtual and actual boundaries such as delimiters of a park or a city.

Road & Railway describes all spatial features which provide a track for moving cars, locomotives, humans etc. from an origin to a destination.

Utility consists of all the information about the public services such as gas, water, electricity, etc.

Structure consists of all the structured spatial data which means all the construction on a given site which is not roofed and walled, such as a bridge, dam, airport runway, etc.

Hypsography includes all the information about the land form and relief.

Properties and relationships among spatial objects

The topographic model is structured according to the object-oriented concept. This concept describes different relationships and hierarchies among the classes and objects. Figure 1 presents part of the proposed model with the relationships among the object classes. The following is the description of the model corresponding to Figure 1. The role of this example is not to present a complete definition but only to illustrate the approach.

class *Road & Railway*

Attributes ;
- OID : integer
- Name : string
- Length : real
- Date-of-construction : date
- Pass-above : list of bridges
- Pass-under : list of bridges
- Pass-through : list of tunnels
- Intersection-road : list of roads
- Intersection-railway : list of railways
- Last-maintenance : date

Constraints ;
- Length > 0
- last-maintenance >= date-of-construction

Class methods ;
- Create
- Delete
- Modify

Object methods ;
- Draw
- .
- .

The class *Road & Railway* has a relationship Pass-under and Pass-above with the object class *Bridge* which means a road or a railway may pass the roads, railways, or rivers via the bridges. It also has a relationship Pass-through with the object class *Tunnel* which means a particular road or railway should be passed through tunnels.

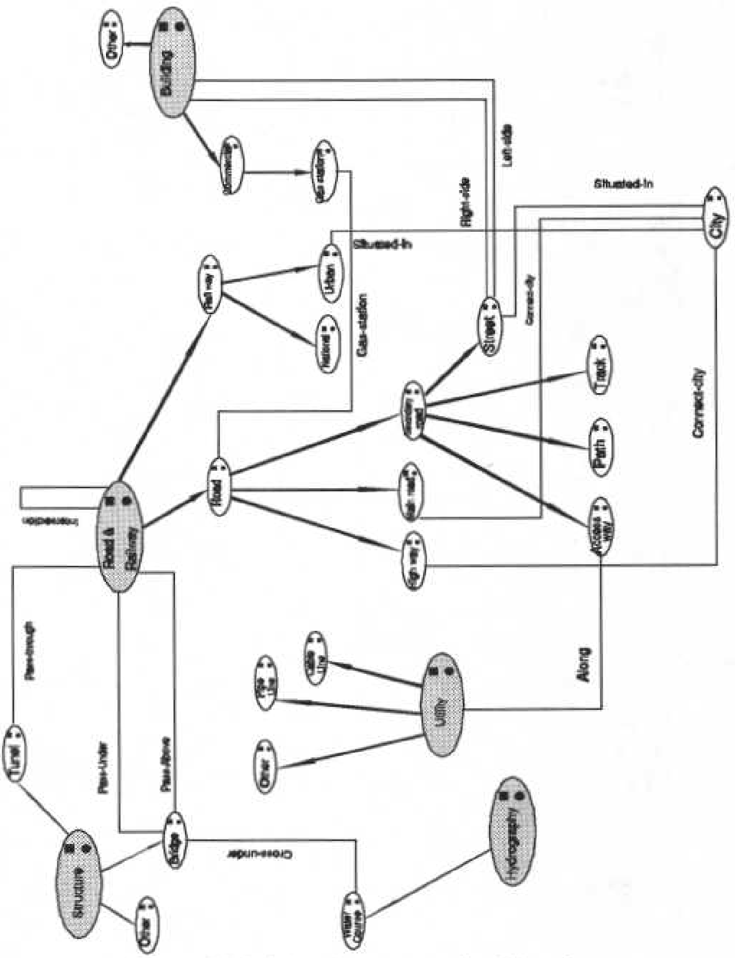

Figure 1: *Relationships among the object classes*

Class *Road* also has its own properties as follows :

class *Road* is a subclass of *Rail & Railway*

Attributes ;
- Width : real
- Surface : string
- Lanes : integer
- Gas-station : list of gas stations

Constraints ;
- Width > 0

Class methods ;
Object methods ;

There is a relation Gas-station among the class *Road* and the class *Gas station building* which indicates all available gas stations along a particular road.

class *Secondary road* is a subclass of *Road*

Attributes ;
Direction : integer

Constraints ;
Direction = 1 or 2

Class methods ;
Object methods ;

class *Street* is a subclass of *Secondary road*

Attributes ;
Situated-in : city
Right-Side : list of buildings
Left-side : list of buildings
Traffic-flow : real

Constraints ;
Class methods ;
Object methods ;

In this research only some parts of the proposed model including the most important aspects of the object-oriented approach are considered and implemented. The practical work is done using Oracle and Arc/Info. PL/SQL and AML are used for the implementation of data types and methods.

Conclusions

Object-orientation is currently the most promising approach to spatial data handling. Among the several ways of introducing these concepts in GIS is implementing object-oriented features by using conventional tools. This can be done for instance by taking existing (conventional) software systems and "emulating" object-orientation by putting an object shell on top of them using available data definition and (macro) language facilities. This research must be seen in the context of a larger research program on intelligent geographical information systems. One of its goals is to investigate alternative ways of implementing topographic databases by using conventional (available) software systems.

References

Bhalla, N. (1991): Object-oriented data models: A perspective and comparative review. In: Journal of information science (17), pp. 145-160.

Bonfatti, F., Cantaroni, R., Gentili, L., Murari, C. (1993): Object-oriented support to the design of Geographic information systems. In: Proceeding of EGIS., Vol. 1, pp. 754-763.

Carey, M.J., Dewitt, D.J., Richardson, J.E., Shekita, E.J. (1986): Object and file management in the EXODUS extensible database system. In: Proceeding of 12th international conference on VLDB, Kyoto, Japan, pp. 91-100.

Date, C.J. (1986): An Introduction to Database Systems. Volume I, Reading, Massachusetts, Addison-Wesley Publishing Company.

Dittrich, K., Dayal, U. (Ed) (1986): Proceedings of the international workshop on object-oriented database systems., IEEE computer society press, Asilomar, California, New york, USA.

Egenhofer, M.J. (1991): Extending SQL for graphical display. In: Geography and Geographic information systems, Vol. 18, No. 4, pp. 230-245.

Egenhofer, M.J., Frank, A.U. (1988): Object-oriented modelling: A powerful tool for GIS. Seminar workbook 'object-oriented technology' presented november 29/1988, San Antonio, Texas, Published by: NCGIA.

Egenhofer, M.J., Frank, A.U. (1989): Object-oriented modeling in GIS: inheritance and propagation. In: E. Anderson (Ed), Proceedings of Auto Carto 9, American Congress on Surveying and Mapping, American Society for Photogrammetry and Remote Sensing, Falls Church, pp. 588-598.

Healey, R.G. (1991): Database management systems. In: Geographic information systems. Volume I, principles, edited by: Maguire, Goodchild, Rhind, Longman, Scientific & technical, Bath press: Avon

Kemppainen, H., Sarjakoski, T. (1990): Object-oriented GIS: A review and a case study. In: Proceeding of ISPRS commission III, China.

Khoshafian, S., Abnous, R. (1990): Object Orientation, Concepts, Languages, Databases, User Interfaces. John Wiley & Sons.

Kim, W., Lochovsky, F.H. (1989): Object-oriented concepts, databases and applications. ASCM press, New York.

ONTOS (1991): ONTOS Developers Guide. Ontos Inc., Burlington Massachusetts, Three Burlington Woods.

ORACLE (1992): PL/SQL user,s guide and references. ORACLE manual, ver. 2.0.

Peuquet, D.J., Marble, D.F. (1990): Introductory readings in geographic information systems. Taylor & Francis.

Stonebraker, M., Rowe, L.A., Hirohama, M. (1990): The implementation of POSTGRESS. In: IEEE Transaction on knowledge and data engineering, 2 (1),pp. 125-142.

Webster, C. (1990): The Object-Oriented paradigm in GIS. In: Proceeding of ISPRS, Commission III, China.

Wessels, C. (1993): Object-orientation and GIS. In: Proceeding of EGIS., Vol. 2, pp. 1462-1471.

Worboys, M.F., Hearnshaw, H.M., Maguire, D.J. (1990): Object-oriented data modelling for spatial databases. Int. J. Geographical Information Systems, Vol. 4, No. 4, pp. 369-383.