# MAPMANAGER: THE DESIGN OF A COM-BASED GIS COMPONENTWARE

Xiaojun Tan [a], Fuling Bian

Wuhan University, School of Remote Sensing and Information Engineering, Wuhan, 430079,China
[a] txj72@263.net

**Commission IV, WG IV/1**

**KEY WORDS:** GIS, COM/DCOM, Spatial Database, Spatial ADT, Structured Storage, Spatial Access Method, Spatial Query, Visualization

**ABSTRACT:**

GIS software development is now moving in the direction of components. The design of a two-dimensional GIS componentware – MapManager, which is based on the COM/DCOM technology, is introduced here. In the process of GIS software design, we should mainly consider the following aspects: spatial database design, visualization, data interface, spatial query and analysis, editing and mapping. At last, we discuss the shortcomings of MapManager and identify the need for future research in the integration with DBMS.

## 1. INTRODUCTION

GIS software and its applications have made great progress since 1960s. Its development could be technically divided into three stages (Fang, *et al.,* 2001b). Because of the complexity of spatial data and the relationships among spatial entities, there were great limitations in the design of early GIS software that was developed in a relatively closed environment. This is the main reason why GIS software was kept far away from the mainstream of information technology (Fang, *et al.,* 2001a). With the maturity of Object-Oriented methodology, a lot of Object-Oriented software development tools, especially component-based development (CBD) tools, are available now. It has already been a tendency of the software technology. This also deeply influences the development of GIS software. The third generation GIS software development is now moving in the direction of components. An important milestone indicates the componentization trend is the OpenGIS project proposed by OGC (Open GIS Consortium), which was founded in 1994. OpenGIS is defined as transparent access to heterogeneous geodata and geoprocessing resources in a networked environment. The goal of the OpenGIS Project is to provide a comprehensive suite of open interface specifications that enable developers to write interoperating components that provide these capabilities (OGC, 1996). The componentization of the major GIS software began in the middle of 1990s, and was almost finished by the end of twenty centuries (Fang, *et al.,* 2001b). Compared with the traditional GIS software, GIS componentware has the following advantages: efficient and seamless system integration, specific programming language unnecessary, popular, and cheap (Song and Zhong, 1998). This paper introduces the design of MapManager – a small two-dimensional GIS componentware.

## 2. DESIGNING MAPMANAGER

There are two important standards for CBD: Microsoft COM (Component Object Model) /DCOM (Distributed Component Object Model), and OMG (Object Management Group) CORBA (Common Object Request Broker Architecture). COM/DCOM takes a lead position in the market and becomes the industrial standard gradually. The ActiveX controls, which are based on the COM/DCOM technology, are the components used most widely in visual programming nowadays. MapManager, a GIS ActiveX control with 41 OLE automation objects and 627 interfaces (including properties, methods and events), is based on COM/DOCM. The design of these objects and interfaces is very important to the functionality of the software. Besides conforming to componentware specifications and providing standard external interfaces, the design of GIS componentware needs the same consideration as traditional GIS software such as: the design of spatial database; visualization of spatial information; data interface with external system; and the design of system functionalities (including editing, querying, spatial analysis, and mapping).

### 2.1 Designing Spatial Database

Spatial database is the core of GIS software. The design of a spatial database takes into account the following four aspects: the representation of spatial data types, internal-memory model of spatial database, external-memory model as well as spatial access method.

Early GIS software used the file system to store the geometric data structures. There has been suggestion and request for the use of database management systems since the early 1980s. It is now being enabled by industry's support of specialized spatial versions of database systems (e.g., Spatial Data Blades and the Spatial Data Option) or middleware (e.g., the Spatial Data Engine). To store the geometric data structures, most commercial relational databases provide long fields (also called binary large objects or memo fields) that serve as simple containers. One of the columns in the relation is declared to have variable length. The geometric representation is then stored in such a long field in a way that only the application programs can interpret it, while the database system itself usually cannot decode the representation. It is, therefore, impossible to formulate or process SQL queries against that column. To implement geographic concepts in a high-level, compact, and reusable way, the use of spatial data types has

become the standard method. Although early attempts were made to rely exclusively on those data types offered by standard programming languages and database systems, it has become common practice to identify spatial data types and to link them with their related operations. Object-oriented methods, with encapsulation and hiding of implementations, have favored this approach. Abstract data types (ADTs) based on object-oriented methods provide a robust way to implement complex data types. The basic idea is to encapsulate the implementation of a data type in such a way that one can communicate with instances of the data type only through a set of well-defined operators. The internal implementation of the data type and its operators are hidden to any external users. They have no way to review or modify those interior features. Object-oriented concepts can easily be adapted to the implementation of spatial ADTs and operators (Egenhofer, *et al.,* 1999). The OGC Technical Committee has released an implementation specification named "OpenGIS Simple Features Specification for OLE/COM Revision 1.1", which has a detailed definition on the geometric class hierarchy (see Figure. 1). The definitions of spatial ADTs in MapManager are based on a subset of those geometric classes, which covers all except GeometryCollection and all its subclasses.

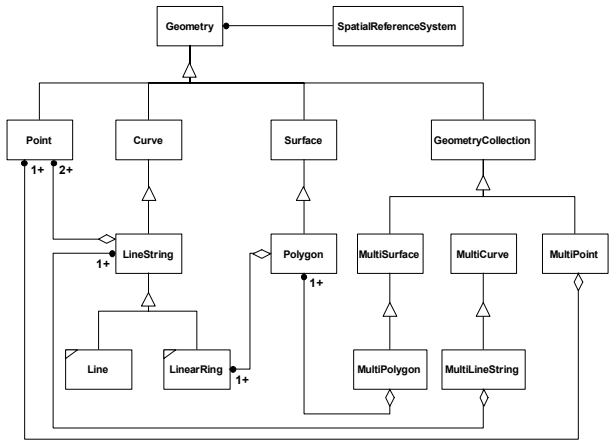


Figure 1. Geometry Class Hierarchy (from OGC 1999a)

The internal-memory model of spatial database refers to how three kinds of information – spatial metrical information, spatial topological information as well as non-spatial information – are arranged in the physical primary memory. All the features in the primary memory are stored in a global object named "Database". There are four spatial data sets, named node data set, arc data set, line data set and polygon data set respectively, and one geo-feature data set in the "Database" object. The spatial information – both metrical and topological – is stored in the four spatial data sets. The node data set stores the node's ID, coordinates, and IDs for all arcs containing this node in anti-clockwise order. The arc data set stores the arc's ID, coordinates list for all points on this arc, and IDs for all lines and polygons containing this arc in anti-clockwise order. The line data set stores the line's ID, IDs for all arcs contained in this line, and IDs for all geo-features containing this line. The polygon data set stores the polygon's ID, IDs for all arcs on the boundaries of this polygon, and IDs for all geo-features containing this polygon.

In the geo-feature data set, semantic information is defined for all geo-features (e.g. road, river, building etc.). There are four kinds of geo-features in MapManager: point feature, line feature, polygon feature and annotation feature. The coordinates of point feature and annotation feature are stored in the geo-feature data set, while those of line feature and surface feature are stored in the above four spatial data sets.

The external-memory model of spatial database refers to how the data is stored persistently in the secondary memory. As stated above, it is a trend to introduce commercial database management system into spatial data management. However, it may be better to use file system for small GIS software like MapManager. Fortunately, the structured storage technology provided by COM/DCOM is a good solution for persistent storage for MapManager. Structured storage is sometimes called "a file system within a file" because it can treat a single file as if it is capable of storing directories and files. A file created using the structured storage service contains one or more *storage*, roughly equivalent to directories, and each storage can contain zero or more *stream*, roughly equivalent to files. A storage can also contain any number of *substorages* (Eddon and Eddon, 2000). The structured storage of MapManager is shown in Figure 2.

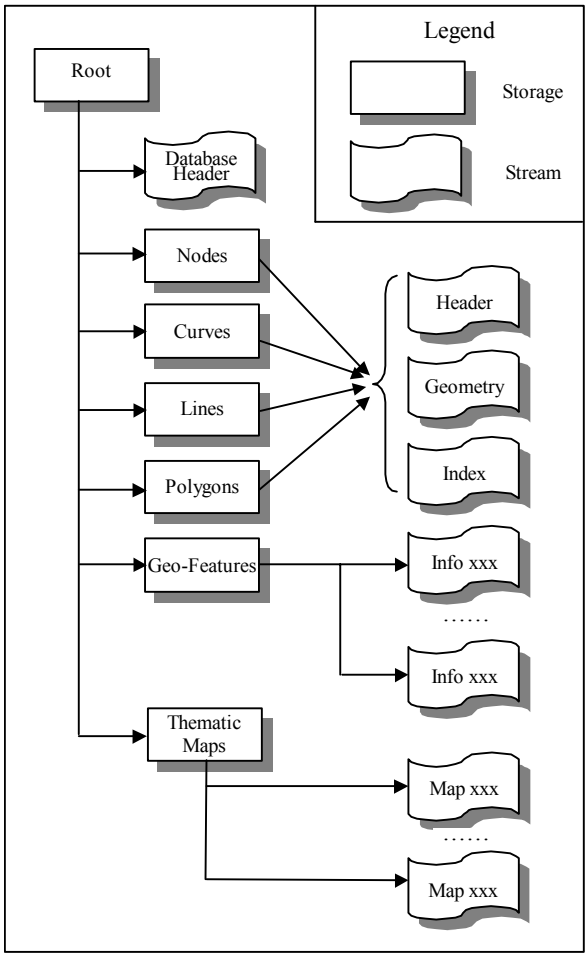


Figure 2. Structured Storage of MapManager

In order to deal with massive data in large application, a feasible spatial access method (SAM), which manages the data exchange between internal and external memory, is needed. The foundation of SAM is spatial index technology. So far there are a lot of spatial index structures such as quadtree, K-D-tree, K-D-B-tree, R-tree, R+-tree, etc. (Samet, 1990; Samet, Aref, 1994; Worboys, 1995; Garcia-Molina, *et al.,* 2000). A common

approach to search spatial objects consists of a two-step process: (1) choosing an approximation (e.g., a simpler shape, such as Minimum Bounding Rectangle) that can be indexed and serves as a fast filter and (2) using the original geometry to assert the retrieval condition only for the initially retrieved objects to eliminate false hits. An index may only administer the MBR of each object, together with a pointer to the description of the object's database entry (the object ID). With this design, the index only produces a set of candidate solutions. This step is therefore termed the filter step. For each element of that candidate set we have to decide whether the MBR is sufficient to decide that the actual object must indeed satisfy the search predicate. In those cases, the object can be added directly to the query result. However, there are often cases where the MBR does not prove to be sufficient. In a refinement step we then have to retrieve the exact shape information from secondary memory and test it against the predicate. If the predicate evaluates to true, the object is added to the query result as well, otherwise we have a false drop (Egenhofer 1999). The spatial index structure of MapManager is square grid, which can manage hundreds of megabyte data stably and efficiently in practice with the help of spatial object's MBR. However, its performance is degraded quickly when the size of database grows.

### 2.2 Visualization

The visualization of spatial information includes the following four aspects: map projection and coordinate system, management of thematic maps and layers, symbolization, and visualization of statistical data. Like the other GIS software, MapManager supports all kinds of common map projections and coordinate systems. The visualization of statistical data could be carried out with the help of other commercial componentware. Here we mainly discuss the management of thematic maps and layers as well as symbolization.

In MapManager, data in spatial database is represented by the means of thematic maps and map layers. There are several thematic maps in a spatial database, and the arrangement of these thematic maps is similar to an atlas. Each thematic map contains several map layers, each of which has a property named "Symbol" and is linked to a kind of geo-feature in the geo-feature data set. When a map is displayed, all the features contained in the map layer are drawn with the proper symbol. All the map layers in a thematic map are organized as a tree-like hierarchy. There are no limitation on the height of the tree and the number of the sub-nodes.

There are four kinds of symbols, point symbol, line symbol, polygon symbol, and annotation symbol, according to four kinds of geo-features. A special program, Symbol Designer, has been written to help designing symbols (see Figure. 3). When a geo-feature is symbolized, the coordinate lists of that feature are passed to the symbol and the symbol itself will finish the drawing procedure. We put forward the concept of "exceptional symbol" to deal with the situation that features of the same type require different symbols. Each feature with an "ExceptionalSymbol" property will be treated differently by the drawing program. The "exceptional symbol" could avoid the conflict between the commonness and individuality of symbols. However, the drawing performance will be degraded if there are too much exceptional symbols.

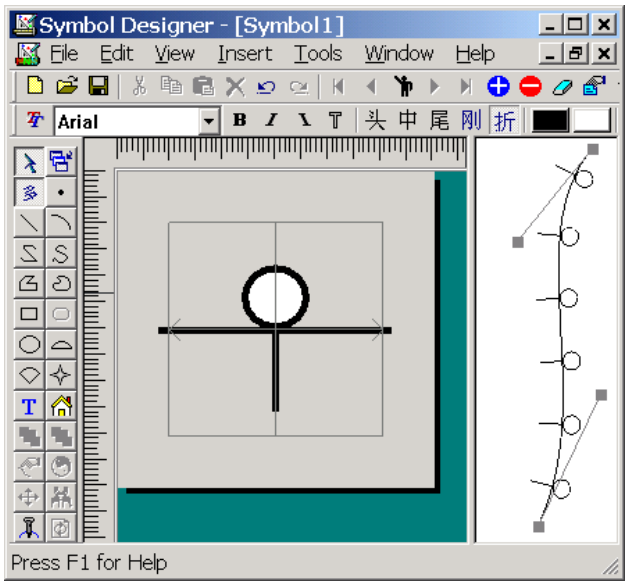


Figure 3. Symbol Designer Program

### 2.3 Data Interface

In terms of data input, MapManager doesn't offer the functionality for digitizing as well as image scanning. However, it provides consistent data interface to import spatial data from outside. This process is reversible – users can also export data to the other system. Currently, MapManager provides four vector data interfaces: Autodesk DXF file, ESRI E00 File, ESRI shape file, and Mapinfo Interchange File.

### 2.4 System Functions

Here we will mainly discuss three aspects of system functions: spatial query, editing and mapping.

For many years, the database market has been dominated by the Structured Query Language (SQL). There has been a long discussion in the literature as to whether SQL is suitable for querying spatial databases. It was recognized early on that relational algebra and SQL alone are not able to provide this kind of support. In some sense, however, with the success of SQL the discussion about its appropriateness has become a moot point. The question is not whether SQL should be used — SQL is and in the foreseeable future will be used to query spatial databases as well. The question is rather which kind of extensions are desirable to optimize user friendliness and performance of the resulting spatial data management system (Egenhofer, *et al.,* 1999). Various extensions to SQL have been proposed to deal with spatial data (Egenhofer, 1992). The ANSI Committee on SQL3 are developing an integrated version of such spatial extensions called SQL/Multimedia (SQL/MM), which is a suite of standards that specify type libraries using SQL's object-oriented facilities. A "quasi-SQL" language, which is an extension to SQL, is used in MapManager to implement the spatial query function. It is based on the definition of spatial relationship predicates. In OGC's implementation specification, there are five named predicates based on the dimensionally extended nine-intersection model (DE-9IM): Disjoint, Touches, Crosses, Within and Overlaps. For user convenience, eight spatial relationships between spatial objects are given as following (OGC, 1999b):

- Equals

- Disjoint
- Intersects
- Touches
- Crosses
- Within
- Contains
- Overlaps

By extending SQL to incorporate these spatial relationships with quasi-SQL language, it is possible to do spatial query efficiently in MapManager.

The usability of GIS software owes a lot to the data edit function. In MapManager, the edit functions are decomposed into a lot of meta-operations, and are realized by the objects' properties, methods, or events. Users can combine these meta-operations to achieve different effects. We pay lots of attention to the automatic maintenance of topological relationship when designing the editing subsystem. And the object-snapping function works very well.

MapManager has powerful mapping capability. There are a whole set of layout element objects that satisfy a lot of applications. The layout could be output to either hard copy devices or standard output formats such as Encapsulated PostScript file.

## 3. CONCLUSIONS

Some professional applications have been developed on MapManager version 1.0, which is proven to be an applicable GIS component. However, there are many shortcomings in the software, and some are listed as following:

- No supporting for complex collection spatial objects
- No concurrency control, because without the help of DBMS
- SAM not efficient enough
- Drawing performance degraded by the hierarchy of symbol object

There are lots of works to do to overcome all these shortcomings. First of all, what we want to research next is to provide accessibility to spatial data in large commercial DBMS.

## REFERENCES


Eddon, G., Eddon, H., 2000. *Inside COM+ Base Services*. Microsoft Press, Redmond, Washington.

Egenhofer, M. J., 1992. Why not SQL! *Int. J. Geographical Information Systems,* 6(2), pp. 71-86.

Egenhofer, M. J., Glasgow, J., Günther, O., Herring, J. R., Peuquet, D. J., 1999. Progress in computational methods for representing geographical concepts. *Int. J. Geographical Information Science,* 13(8), pp. 775-796.

Fang, Y., Tian, G. L., Shi, Z. Z., Zhou, C. H., 2001a. Modert Itand 4th GIS software. *Journal of Image and Graphics,* 6A(9), pp. 824-829.

Fang, Y., Zhou, C. H., Jing, G. F., Lu, F., Luo, J. C., 2001b. Reasearch of 4th Generation GIS software. *Journal of Image and Graphics,* 6A(9), pp. 817-823.

Garcia-Molina, H., Ullman, J. D., Widom, J., 2000. *Database System Implementation,* Prentice Hall, Inc.

OGC, 1996. The OpenGIS™ Guide. http://www.opengis.org/techn/guide/guide1.htm

OGC, 1999a. OpenGIS® Simple Features Specification for OLE/COM. http://www.opengis.org/techno/specs/99-050.pdf.

OGC, 1999b. OpenGIS® Simple Features Specification for SQL. http://www.opengis.org/techno/specs/99-049.pdf.

OGC, 1999c. The OpenGIS™ Abstract Specification – 99-AS-RFP009. http://www.opengis.org/techno/abstract/99-AS-RFP009.pdf.zip.

Samet, H., 1990. *The Design and Analysis of Spatial Data Structures,* Addison-Wesley.

Samet, H., Aref, W. G., 1994. Spatial data model and query processing. *Modern Database Systems: The Object Model, Interoperability, and Beyond.* Addison Wesley/ACM Press, Reading, MA, pp.338-360.

Song, G. F., Zhong, E. S., 1998. Research and Development of Components Geographic Information Systems. Journal of Image and Graphics, 3(4).

Worboys, M.F., 1995. *GIS: A Computing Perspective,* Taylor & Francis Ltd, London