A SCALABLE, OBJECT-ORIENTED GIS FRAMEWORK*

Mitrović Antonija
Computer Science Department, University of Canterbury
Christchurch, New Zealand

Mitrović Dejan
Present address: Trimble Navigation
Christchurch, New Zealand

Djordjević-Kajan Slobodanka
Petković Milan
Computer Science Department, University of Niš
Niš, Yugoslavia

ABSTRACT

This paper presents the underlying philosophy and design rationale of *GinisNT*, a scalable, OO framework for developing GIS applications. The fundamentals of *GinisNT* is an OO data model and an OO application development methodology. *GinisNT* OO data model supports all important concepts found in OO theory and is actually implemented on top of the relational data model. *GinisNT* automatically maps between the two data models, thus making the usage of a RDBMS transparent to the user, who can concentrate on the application itself. While developing the user interface, we have taken into account existing knowledge and ways of handling spatial data of our prospective users, so no training is needed for the users to start working with *GinisNT*. Furthermore, no additional funding is required also, as the framework is based on cheap hardware platforms and existing software infrastructure (RDBMS). *GinisNT* is currently under development, as well as the first GIS application based on it, which is a telephone network maintenance system. We believe that the sound foundation of the framework will ensure fast development of high-quality GIS applications.

*Presented at the International Society for Photogrammetry and Remote Sensing "Workshop on New Developments in Geographic Information Systems," Milan, Italy, 6-8 March 1996.

1.0 INTRODUCTION

Lower costs of computer hardware and general awareness of effectiveness and usefulness of GISs have had valuable impacts on their wider application. However, numerous problems still remain, especially in the development of end-user GIS applications. GIS software is very complex and the tools for developing applications, most often in the form of macro languages, typically have restricted functionality. Data models on which GIS software is based have been developed with spatial data in mind, and are not readily extendible to support other real-world problems. Numerous legacy systems should be coupled with GIS systems, which present additional difficulties in developing end–user applications.

The research group at the Computer Graphics and GIS Lab at the University of Niš, Yugoslavia has been developing GIS software for eight years now. One of the research directions pursued has been the development of GIS architecture suitable for the implementation of end-user GIS applications under very limited resources. *GINIS* (Mitrović 1993, 1994c; Djordjević 1995a) is a GIS toolkit based on extended relational data model which has been successfully used in several applications (Mitrović et al, 1994a, 1994b, 1995a; Djordjević 1995b). However, the development of new applications on top of *GINIS* is still quite time–consuming and requires detailed knowledge of the toolkit.

This paper presents a novel, scalable, object–oriented GIS framework, referred to as *GinisNT*. In the next section we explain the foundations of *GinisNT*. The basic concepts include the object model, which subsumes an OO data model, OO application development methodology and the supporting tool, and extensible class library. We present the architecture of the framework in section 3, implementation details in section 4 and conclusions in section 5.

2.0 FOUNDATION OF *GinisNT*

One of the inspiring factors for this research was the inadequacy of classical GIS architectures, such as *GINIS* and commercially available GIS environments. They are invariably developed with some sort of spatial applications in mind, so only certain spatial data types and operations are predefined and the tools for end-user application development are not flexible enough. Most of these environments are closed; there is no opportunity for the application developer to extend them and add some desired functionality. Therefore, the development of a spatial application can be very difficult if it is substantially different from those supported by the environment used.

The other factor was the resources that were available. In developing countries such as Yugoslavia, not only funds are highly restricted, but there are also additional problems, like inadequate infrastructure, instability and lack of educated people. In such settings, GIS software must run on cheap hardware platforms which may even be shared with other software. It is often the case that a single computer will be used for a GIS application and also for some other applications, like resource management. Furthermore, data should be easily interchangeable between GISs and other applications.

New spatial data sources (GPS measurements, satellite remote sensed data, available spatial databases) should be used for data acquisition. The level of data accuracy should be chosen as to satisfy the requirements of applications, but not to impose too high a demand on existing resources, since increasing data accuracy will result in exponential increase of required resources.

2.1 OBJECT ORIENTATION AND *GinisNT*

Starting from the above requirements (Mitrović 1995b), the goal of this research was the development a GIS framework which would enable extensibility of developed applications and databases. The cost of GIS applications development is very high, and therefore extensibility and reusability of developed software have the highest priorities. Taking into account all the previous factors, the object-oriented paradigm was chosen as the most natural one. It is logical to represent spatial objects as objects in the application. OO framework is highly expressive and easy to understand. The problem arises with the need for persistent storage of objects. The main difficulty in coupling object–oriented applications with databases is the impedance mismatch problem: transformation between representations used in OO application and database. This transformation may in some cases result in loss of information if the resulting model is not as expressive as the starting one; additionally, it may be the case that some information is hidden by different concepts of the final model. Even if we do manage to represent all aspects of one model using the concepts available in the other one, time is needed to restructure the information represented.

Two different approaches have been distinguished in persistent storage for objects (Lee & Weiderhold 1994). If data is stored in an OO database, the problem of impedance mismatch is avoided. This approach is known as *direct object storage*. However, OO databases still suffer from a number of difficulties, such as view integration, indexing and query optimization.

The other approach is *indirect base relation storage*, which couples an OO application with the relational database. The impedance mismatch problem becomes significant here, but, on the other hand, RDBMS are widely used and hardware and software platforms needed are already available. The problems of sharable concurrent access to data is solved, and the growth of the system is supported. These are the reason in favour of the indirect base relation storage approach for practical applications, especially taking into account the estimate that the current market for OODBMS is less than 1% of the total database market (Kim 1993). The relational data model is the prevalent data model today and it will be the dominant one for at least the next decade.

The indirect base relation storage approach can be realized from two perspectives: *object-centered* and *relation-centered* (Lee & Weiderhold 1994). The difference between the two comes for the primary source of data. In the object-centered perspective, relation schemas are generated from class descriptions. In the latter one, it is assumed that relation schemas already exist, and class descriptions are derived from them. As pointed out by Wiederhold, the query used to instantiate an object from corresponding relations can be quite incomprehensible for users because of the normalization process applied to the relations. Object-centered perspective gives more freedom to the programmer. GinisNT is one realization of this approach (Mitrović et al. 1995b). The following subsection describes its basic concepts.

2.2 BASIC CONCEPT OF *GinisNT*

The basic concepts of *GinisNT* include *GinisNT* object model, comprised of an OO data model, OO methodology and an extensible class library; spatial database, project (spatial application) and spatial feature.

A **spatial database** is a set of related data describing various aspects of spatial and non-spatial objects of interest, including metadata describing the structure of the database itself. Spatial databases are implemented using the *GinisNT* **OO data model**, which supports all the basic concepts found in OO theory, such as objects, classes, object identity, inheritance, aggregation, association and version control. The classes in the OO schema for metadata repository correspond to the concepts of the OO data model, as illustrated in figure 1.

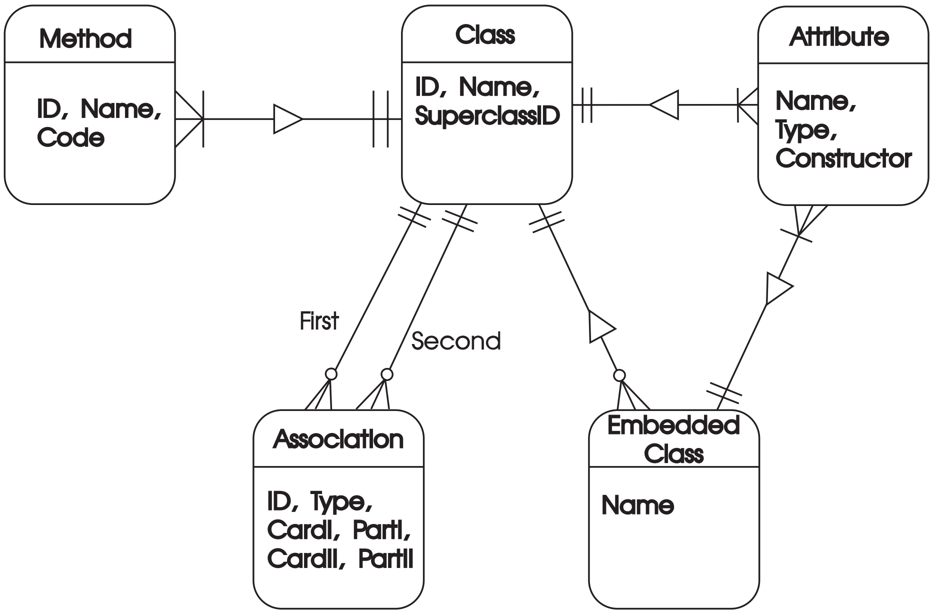

Figure 1: Structure of the Metadata Repository

A **project** is a spatial database subset corresponding to particular GIS application being developed. The project model defines which entities are used, associations between them, the methods defined and types of analyses available.

The **class library** serves as the staring point for developing application classes. It consists of four groups of classes (figure 2): metadata classes, used for describing the structure of the GIS application database, application-building classes, feature and attribute hierarchies. The first two groups of classes are fixed and present the framework for application development. The feature hierarchy consists of a number of predefined classes for modeling of both spatial and non-spatial objects. The attribute hierarchy supports three types of attributes, simple, composite and multivalued. These hierarchies are extensible, thus enabling the user to specify application–specific objects, their characteristics and behaviour.

A **feature** is an abstraction of all the entities of the same type. *GinisNT* provides definitions of the primitive types (aspatial and spatial ones, such as line, area and grid geometrical features, or node, chain, polygon and DEM topological ones) at the same type enabling the user to define new ones by inheriting the attributes of the existing classes. The feature hierarchy is depicted in figure 3.

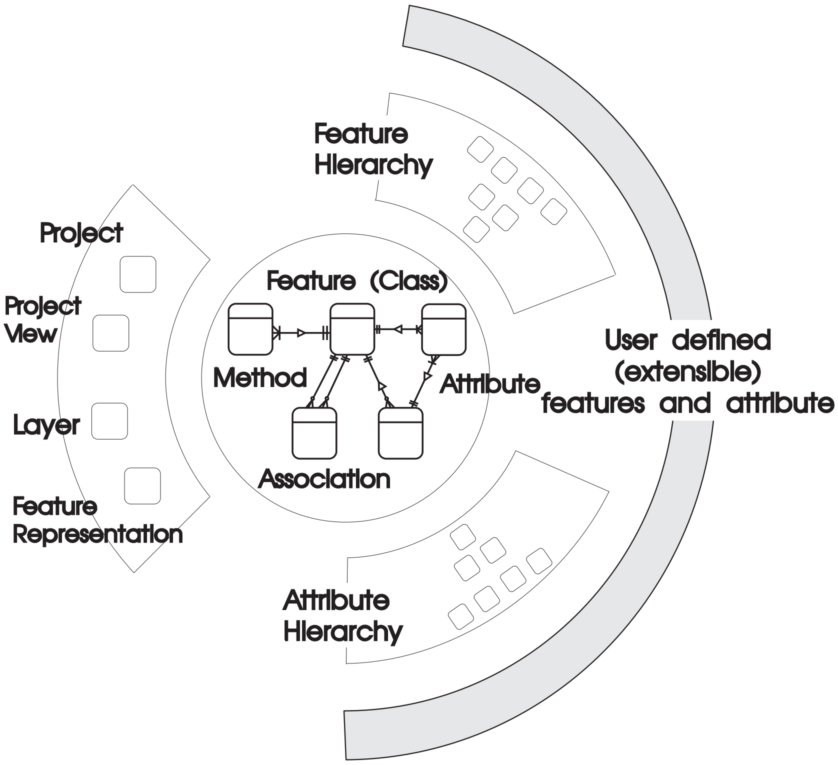

Figure 2: Class Library

3.0 *GinisNT* ARCHITECTURE

GinisNT consists of four components: Object Wizard, Mediator, Spatial Data Manager (SDM) and user interface, as illustrated in figure 4. Object Wizard is a CASE tool which implements *GinisNT* OO methodology by supporting schema definition. It enables the user to create an application by defining its OO model and corresponding classes. This component allows the user to create the application by defining its OO model and corresponding classes. The methodology for developing applications in *GinisNT* consists of the following phases:

1. Creating a project
   Basic project data is defined, like date/time of creation, the area of interest, the name of the developer, access rights for other users and so on.

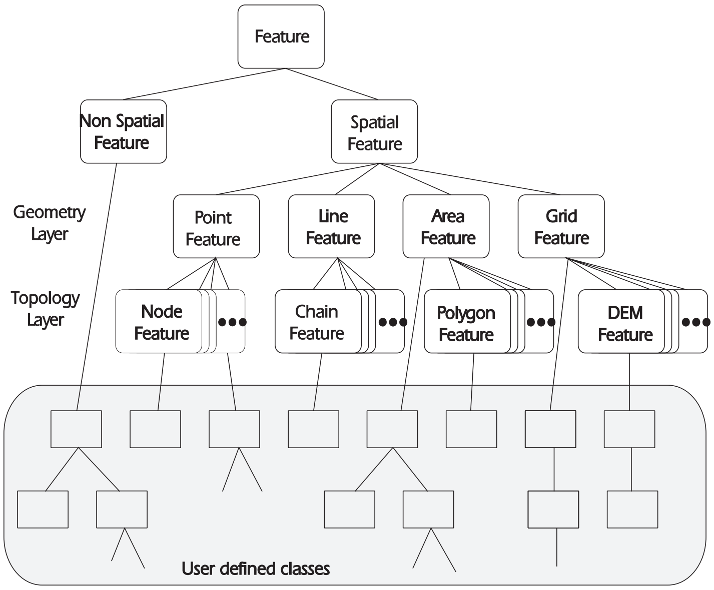

Figure 3: Feature Hierarchy

2. Creating the OO model for the project
   Some of existing features are selected and new ones are defined. Object Wizard enables the user to define project classes on the basis of the predefined class hierarchy, and using attributes from the attribute hierarchy or by defining new attribute types. It also facilitates the definition of associations and methods.

3. Defining operations (spatial or non-spatial data analyses) which can be performed on the subset of the spatial database. The operations are defined as methods for the project.

4. Saving the project definition in the metadata repository, mapping it to the relational data model and creating corresponding relations.

The first three phases are the responsibility of the project developer, while the last one is completely automated and invisible to the developer. Since the OO data model is implemented on top of relational DBMS, Object Wizard maps the application OO model into the relational schema, creates relations and stores the definition of the application in the meta-data repository. In such a way the relational DBMS used at the implementational level is made transparent to the user.

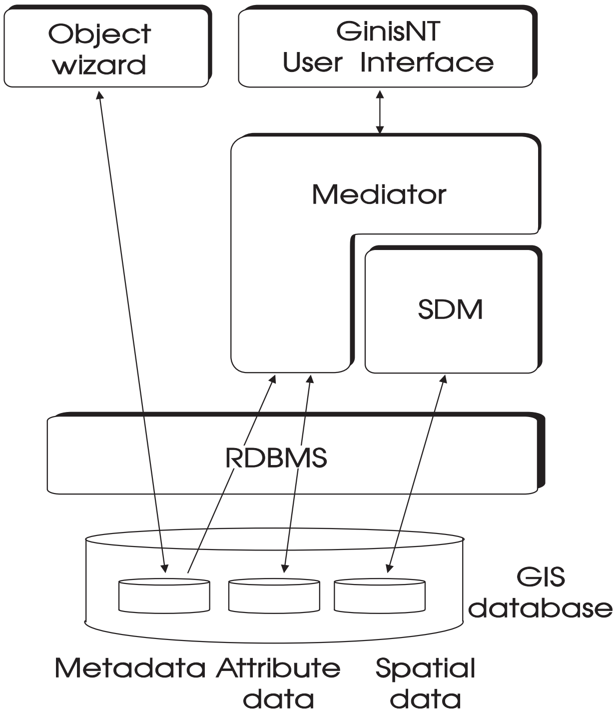

Figure 4: The Architecture of *GinisNT*

Mediator serves as an intermediary between the OO application and the relational database. It provides run-time support by processing user's demands. When a user requires some operation to be performed, Mediator retrieves appropriate information from the metadata repository to decide which relations to access in order to locate object's data or methods. It generates database statements and provides the user interface with the data necessary to customize its appearance. Mediator also instantiates objects from relations. As discussed earlier, the specification of the way (view) to instantiate objects from relations can be incomprehensible for users, since the data about one object may be spread over several tables. The Mediator performs the instantiation without any interference from the user, automatically, using the information in the meta–data repository.

Mediator uses services of the **Spatial Data Manager**, which provides spatial data storage and retrieval in a RDBMS using abstract data types and spatial indexing. A modification of the OneKey/FourKey method (Abel 1983, 1984) is used for spatial indexing. It linearizes the two-dimensional problem of indexing, thus enabling the usage of B+ trees provided by the RDBMS. Since only an approximation of a spatial object is used for indexing purposes, SDM uses geometrical filters. SDM generates a spatial index entry for each new object, processes spatial and combined queries by generating appropriate parts of the query statements and provides geometrical filters.

GinisNT user interface (figure 5) is a collection of user interaction tools which can be adapted to suit the needs of a specific GIS application. An additional function of the user interface is manipulation with raster maps, which are used as visual context in *GinisNT*. Usage of raster maps as backgrounds for the interface has several advantages, some of which are the efficiency of operations, ease of use for end-users already familiar with their paper equivalents, and support for data entry ("head-up" digitizing). MapEdit is a utility program which supports the operations with raster maps. Some of its functions include filtering, deskewing, joining or separation of map segments (figure 6).

4.0 IMPLEMENTATION

We have chosen MS OLE 2.0 (Object Linking and Embedding) as the basis for implementing *GinisNT*. There are many reasons in favour of such a decision; it is the most widely used OO environment which can be run on inexpensive hardware platforms. Furthermore, many application of our end-users already support this standard, which enables easy integration of *GinismT* into existing environments and data exchange. Finally, there are cheap and popular development tools which can be used for the development of end-user applications.

The Windows Component Object Model (COM) provides a means to address problems of application complexity (splitting the application into separate run-time components) and evolution of functionality over time. COM defines and implements mechanisms that allows components to connect to each other as software objects, collections of related methods and the methods' associated state. From this viewpoint, Microsoft OLE is a very appropriate environment for the scalable OO architecture of *GinisNT*. OLE 2 is primarily a set of standard interfaces by which components integration is achieved. Components provide and/or use each others' services through these standard interfaces.

The main components of *GinisNT*, UI, Mediator and SDM are implemented as separate components with predefined (OLE2 standard) interfaces. These components appear to the end-user as a single application. The user can extend *GinisNT* by adding

Figure 5: The Interface of *GinisNT*

new, user-specific components. It is planned for Wizard to be integrated (using OLE2 standard interface again) with MS VisualBasic development environment, thus enabling the user to link his/her own component with predefined *GinisNT* components. OLE2 standard interface is also used as a vehicle for integrating *GinisNT* with other user applications providing easy data transfer and operation flow.

5.0 CONCLUSIONS

The GIS framework presented in this paper is an object-oriented one. It supports extensibility of developed GIS applications and databases. *GinisNT* appears to be

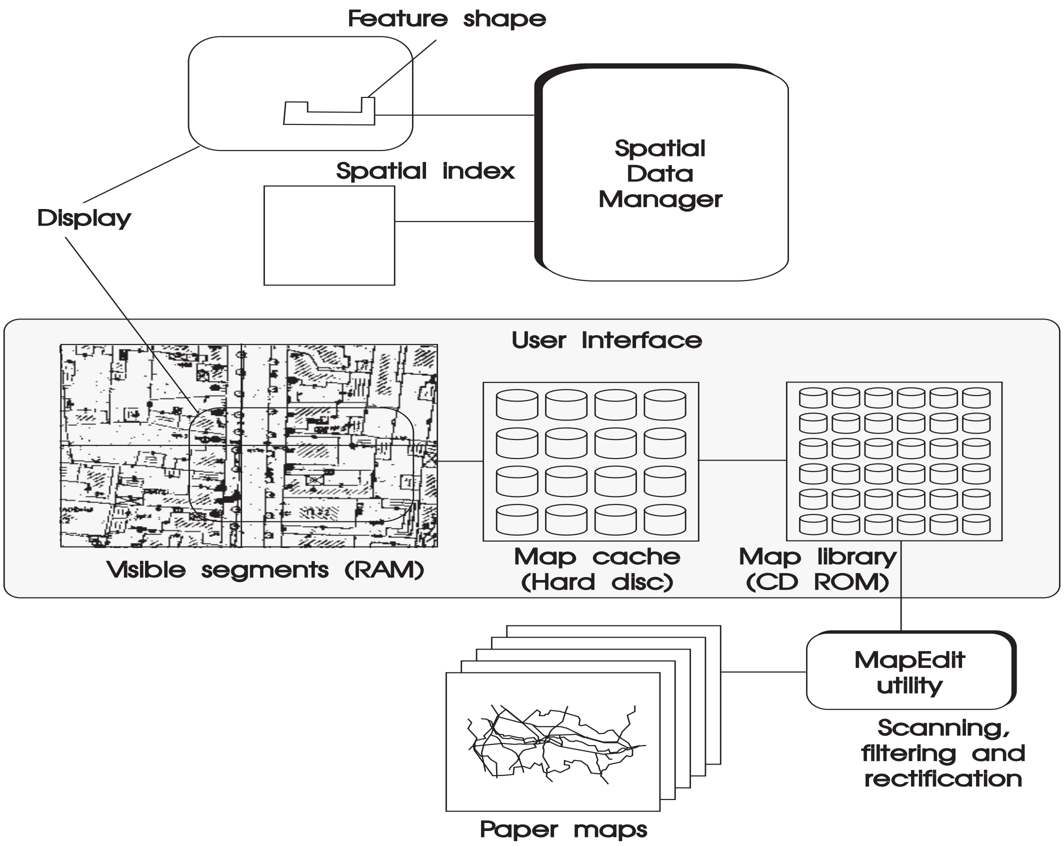

Figure 6: Raster Map Manipulations

completely object-oriented to its users, although it is actually implemented on top of a relational database. OO methodology and data model adopted in *GinisNT* are easy to understand. The system automatically maps OO schema into the relational one and vice versa, therefore making the existence of an underlying RDBMS transparent to the user. *GinisNT* supports the application development process by providing predefined hierarchies of classes. Existing hardware and software resources are maximally reused and there are no needs for additional training.

GinisNT is currently under development, as well as MapPhone, the first application based on it, which is a telephone network maintenance system. Such concurrent development of the framework and the application based on it will allow us to test the fundamentals of the framework and to make changes if necessary. We believe that the sound foundation of the framework will result in fast development of high-quality GIS applications.

6.0 REFERENCES

1. D.J. Abel, J.L. Smith, "A Data Structure and Algorithm Based on a Linear Key for a Rectangle Retrieval Problem." *Computer Vision, Graphics and Image Processing*, Vol. 24, No. 1, pp. 1-13, 1983.
2. D.J. Abel, J.L. Smith, "A Data Structure and Query Algorithm for a Database of Areal Entities." *Australian Computer Jornal*, Vol. 16, No. 4, pp. 147-154, 1984.
3. S. Djordjević–Kajan, D. Mitrović, D. Rančić, M. Petković, A. Mitrović, "MapEdit - Software for Creating a Continuous Raster Map Base." In *Proceedings of EUROCARTO 13*, Ispra, Italy, 2-4 October 1995a.
4. S. Djordjević–Kajan, Mitrović, D., Mitrović, A., Stoimenov, L. Stanić, Z., "Intelligent GIS for Fire Department Services." In *Proceedings of EUROCARTO 13*, Ispra, Italy, 2-4 October 1995b.
5. B.S. Lee, G. Wiederhold, "Outer joins and filters for instantiating objects from relational databases through views." *IEEE Trans. Know. Data Engineering*, Vol. 6, No. 1, pp. 108-119, 1994.
6. W. Kim, J. Garza, A. Keskin, "Spatial data management in database systems: research directions." In *Advances in Spatial Databases*, ed. D. Abel, Springer-Verlag, Berlin, 1993.
7. D. Mitrović, *An extension of RDBMS for application in GIS*, Master's thesis, Computer Science Department, University of Niš, 1993 (in Serbian).
8. D. Mitrović, Z. Radenković, A. Mitrović, "PLEX - a knowledge-based system for urban planning." *Int. J. Systems Research & Information Science*, Vol. 6, pp. 141-149, 1994a.
9. D. Mitrović, S. Djordjević, A. Mitrović, Z. Radenković, L. Stoimenov, "MapSys - a spatial objects control system." In *Proceedings of 3rd International Conference on Automation, Robotics and Computer Vision*, Singapore, WA6.1, pp. 93-97, November 1994b
10. D. Mitrović, S. Djordjević, "An open GIS architecture for inexpensive hardware platforms." In *Proceedings of 10th AM/FM Europian conference*, Heidelberg, Germany, pp. 185-189, 1994c.
11. D. Mitrović, S. Djordjević-Kajan, A. Mitrović, "Intelligent GIS as an aid in urban planning." In *Proceedings of 17th International Cartographic Conference*, Barcelona, Spain, pp. 1486-1490, 1995a.
12. D. Mitrović, A. Mitrović, S. Djordjević-Kajan, D. Rančić, "A GIS solution for developing countries." In *Proceedings of AURISA-95*, Melbourne, Australia, pp. 262-271, November 1995.