INFORMATICA, 2000, Vol. 11, No. 3, 269–280

Intelligent GIS: Architectural Issues and Implementation Methods

Viktoras PALIULIONIS
Institute of Mathematics and Informatics
Akademijos 4, LT-2600 Vilnius, Lithuania
e-mail: vikpal@ktl.mii.lt

Received: September 2000

Abstract. The paper presents an intelligent GIS architecture that enables us to extend GIS functionality by using domain specific knowledge and inference engine. In this architecture, an intelligent agent monitors events, which occur in the GIS environment, and execute tasks depending on user's actions. The intelligent agent includes an expert system shell and knowledge base. A hybrid knowledge representation method is used that integrates rule-based, object-oriented, and procedural knowledge representations.

Key words: intelligent GIS, software integration, software architecture, expert systems.

1. Introduction

Geographic information systems (GIS) are a technology designed to capture, store, manipulate, analyze, and visualize spatially referenced data. GIS are being increasingly used for performing spatial analysis and modeling, moving into areas for which they were not necessarily intended. Integrating GIS with intelligent system technologies can significantly extend possibilities of GIS. A number of works discusses the use of intelligent technologies (knowledge-based systems, artificial neural networks, fuzzy logic systems, etc.) in various GIS application domains.

Knowledge-based technology is used for constructing expert systems. Expert systems combine experience, intuition and judgment to solve ill-structured problems where knowledge and data may be incomplete and/or imprecise. Expert systems can be used in many GIS application domains. A number of examples from different domains of natural resources and environmental management are presented in (Fedra, 1995). Such systems combine models, GIS, and expert systems in a number of customized implementations for specific decision support problems. Expert systems are often used to help configure models and estimate parameters. A number of these "intelligent front end systems" or model advisors have been developed in the environmental domain (Fedra, 1993).

The knowledge-based technology can also be useful in implementing spatial intelligent agents that would help to solve such problems as locating and retrieving spatial information in large networks (and specifically the Internet), facilitate the handling of a

GIS user interface, implementing improved spatial tasks and creating interfaces between GIS and specific software packages (Rodrigues *et al.*, 1995).

Artificial neural networks (ANN) have been used in GIS for over decade. Applications include classifying remote sensing data (Miller *et al.*, 1995), vegetation and land cover mapping (Fitzgerald and Lees, 1996; Foody, 1997), ore reserve estimation (Wu and Zhou, 1993), and modeling spatial interaction (Openshow and Turton, 1994). AAN have several advantages when used as classifiers of complex geographic and remotely sensed data. They normally require no assumptions on the data distribution and can be trained with relatively small sample sets.

Fuzzy logic has been widely used to handle imprecision and to reason about spatial concepts (e.g., near, high, etc.) in text-based queries (Wang, 1994). Altman (1994) introduced fuzzy spatial relations and used them in the spatial analysis.

All these technologies are useful to make GIS more intelligent and powerful. However an intelligent GIS is not only a powerful toolkit. According to (Andrienko and Andrienko, 1999), an intelligent GIS should assist the user in the analysis, and this presupposes the following capabilities: the capability to understand user's information-seeking goals, the capability to select and visualize appropriate data in a way productive for achieving these goals, and the capability to support the user's analytical activity by using of the generated presentations. None of the available GIS possesses such features.

In spite of a number of examples how to use intelligent system technologies in GIS, there is no general methodology how to implement such systems. This paper overviews the existing methods of integrating GIS and intelligent system technologies, and proposes an intelligent GIS (IGIS) architecture that enables us to extend GIS functionality by using domain specific knowledge and reasoning mechanisms. In this architecture, an intelligent agent monitors events, which occur in the GIS environment, and executes tasks depending on user's actions. An example of the prototype system that demonstrates IGIS implementation methods is described.

2. Methods of Integrating GIS with Intelligent System Technologies

Various levels of integrating GIS and intelligent system technologies are possible, ranging from the simple exchange of data files (loosely coupled approach) to a complete integration within one common environment, framework, and user interface (tightly coupled approach).

In tightly coupled systems, intelligent components can be developed directly within GIS using a GIS-based programming language. This effectively extends the functionality of GIS and provides seamless integration from a user interface viewpoint. However, the languages in most GIS are not powerful enough to handle knowledge representation or reasoning mechanisms. Besides, the speed of calculation is rather low because programs are usually interpreted by the GIS.

Loosely coupled systems interact by using files to exchange data between a GIS and other systems. An intelligent system application may read some of its input data from

GIS files and produce some of its output in a format that allows processing and display with GIS. It requires little, if any, software modifications but the speed of data transfer between the expert system and the GIS is low and it is difficult to implement a common user interface (for example, point-and-click operations).

To integrate independently developed software systems, wrappers are often used. A wrapper is a kind of software that is used to attach software components one to another. The wrapping software envelops and provides a point of access to the integrated software. It may encapsulate a system to make it usable in some new way that the unwrapped system was not. Wrappers are common components in a number of software architectures used for integrating software (Ishikawa *et al.*, 1997; Papakonstantinou *et al.*, 1995).

Recent GIS implementations have begun to support the use of client-server techniques as an effective means of transferring data while retaining data structure. This has provided the possibility of linking other systems with GIS by using these techniques. In (Sandhu and Treleaven, 1996), a co-operative approach is described, based on client-server technologies. The authors present two types of this approach. First, with direct co-operation, systems being integrated are directly linked via interapplication communication. Secondly, indirect co-operation is characterized by the existence of an intermediate interface between systems.

In this paper, we propose an architecture, in which the client-server technique is used to organize the interaction between GIS and the intelligent agent that performs various intelligent tasks.

3. Architecture of Intelligent GIS

In this section, we overview the proposed architecture of the intelligent GIS. In this architecture (Fig. 1), the Intelligent Agent monitors events occurring in the geographic information system and responds to them. This is performed by running scenario models that are stored in the knowledge base of the Intelligent Agent. In some works scenarios are also called use cases, system operations or business processes (Jacobson, 1994). We define a scenario model as a knowledge-based program that performs a certain task.

The main features of proposed scenario models are as follows:

- To represent domain specific knowledge, a hybrid representation method is used that integrates declarative (production rules, facts), object-oriented (classes, objects), and procedural (functions) knowledge representations. Therefore, scenario models can be used to integrate qualitative and quantitative reasoning.
- The scenarios models process events occurring in the system. It enables creating an interactive system that responds to user's actions (e.g., clicking a mouse button in the map window).
- The scenario model can manipulate GIS objects by invoking their methods, getting or setting their attributes. It can dynamically change the interface and fit it to a specific task. The scenario model has access to the descriptions of layers and can

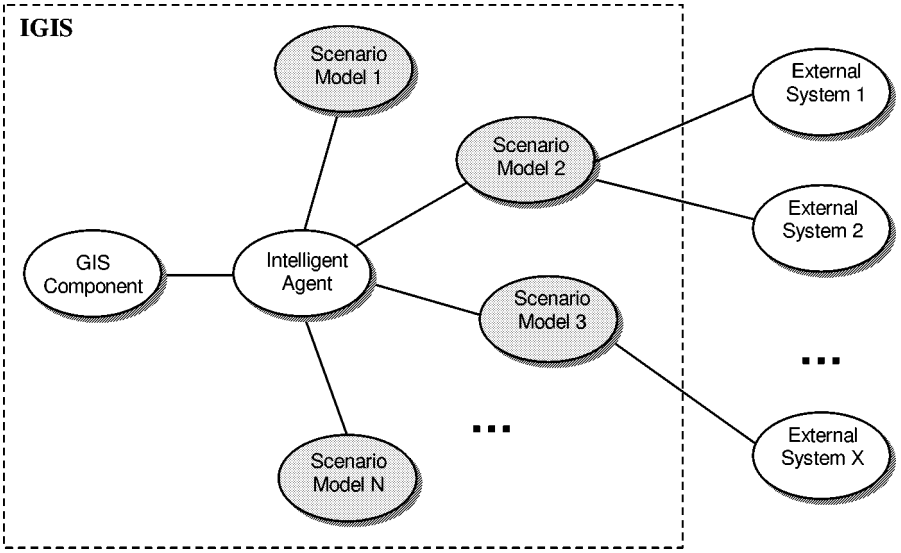

Fig. 1. The overall architecture of intelligent GIS.

change them. It can ask for data from a layer's object. It can also create new geoobjects and change their state dynamically. These changes are immediately displayed in the map window. Besides, the scenario model can automate repeating actions and lead a user step-by-step through a task, often an especially complex or tricky one.

- The scenario model can communicate with external systems. Therefore it can help to implement the interface between GIS and various intelligent systems (e.g., artificial neural networks, fuzzy logic systems), domain specific models, and GIS data processing applications.

These features make it possible to create scenario models to perform a great variety of tasks. Some examples are presented in Fig. 2. In this figure, one scenario model performs some qualitative reasoning task, the second one perform a classification of remote sensed data using an artificial neural network, the third one is responsible for fuzzy spatial querying using an external fuzzy logic system, and the fourth scenario model used for automatic vehicle monitoring. It is important that in our approach the functionality of the GIS application is extended dynamically, without reprogramming and recompiling the program. The knowledge base of the Intelligent Agent can be created incrementally. Scenario models can be created by domain experts or advanced users. In this way the knowledge is accumulated and passed to less experienced users.

Consider the structure of IGIS architecture components more in details (Fig. 3).

3.1. *GIS Component*

The GIS component provides a graphical user interface and a number of tools, including typical GIS tools such as zoom, pan, point query, attribute filters, distance calculations

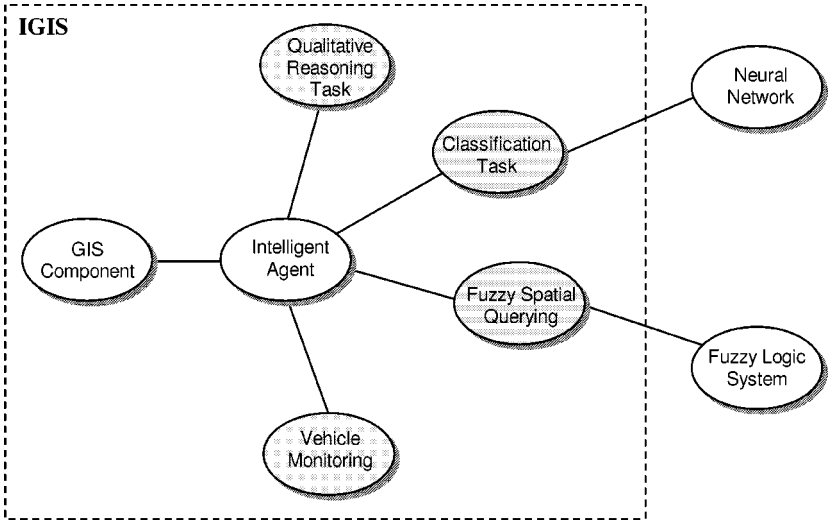

Fig. 2. Examples of scenario models.

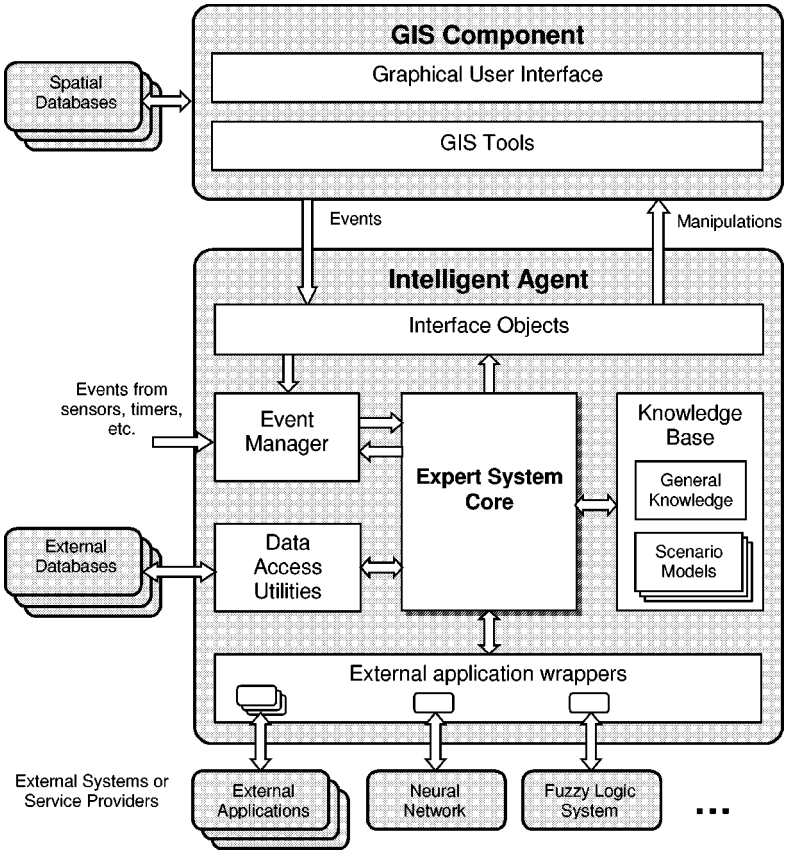

Fig. 3. The architecture of intelligent GIS.

and others. In addition, the GIS component provides interfaces to access its internal objects from external applications. For example, an external program can activate or zoom a map window by manipulating it as an object. Similarly, it can create, show or hide a layer, move a geoobject on the map. It is possible to extend the user interface by adding new commands, toolbar buttons, menu items, and handling events. External applications manipulate objects by using methods, properties, and events associated with the objects.

Important elements of the proposed architecture are map layers and dynamic geoobjects. Layers are responsible for acquiring, constructing, and rendering their own graphical data. Layers can refer to various data sources and enable access to spatial data files and databases of different format. The proposed architecture combines traditional GIS data organization in layers and the modeling of dynamic geoobjects. Dynamic geoobjects are rendered over a static map, which is built up of selected layers. The dynamic geoobject manager enables one to dynamically create and delete geoobjects, change their state (their position, shape or attributes) that can be represented immediately in the map window.

The GIS component can also run independently, without the Intelligent Agent. In this case, it provides basic GIS functions only. When integrated with the Intelligent Agent and domain specific knowledge bases, the system acquires new possibilities.

3.2. *Intelligent Agent*

The Intelligent Agent is composed of the following components:

- *Knowledge Base*. In the IGIS architecture, the Knowledge Base contains general GIS domain knowledge and specific scenario models. The general knowledge describes general constructs, classes, and rules that can be used in scenario models. A scenario model includes specific domain knowledge needed to solve a concrete task. A scenario model is created using general GIS knowledge and expert knowledge of the domain. There is no exact boundary between general knowledge and scenario models. However, their separation enables the reuse of the general knowledge. It is expedient to store class descriptions like *Geometry*, *Geoobject*, etc., in the general knowledge base, and create their specific subclasses in scenario models.
- *Expert System Core*. The Expert System Core implements an inference engine that executes scenario programs. It provides the hybrid knowledge representation language that integrates declarative, object-oriented, and procedural knowledge representations. It also provides a library of predefined functions and possibility to write external functions.
- *Event Manager*. The Event Manager organizes and controls the event queue and passes events to appropriate event handlers. Events occur because of user's actions (clicking a mouse button, selecting a geoobject on the map, selecting a menu item, etc.), after changes of the state of objects (e.g., activating a program, opening a file) or signals from various devices (sensors, timers, etc.). Each event has a time stamp and additional data depending on the type of the event. After an event occurs, the appropriate rules of the active scenario model fire. The right sides of the rules

specify some actions, e.g., select another scenario model, invoke a function, create a fact or object, invoke some object's method. For example, when a user clicks the button of some task, the rule fires that selects the scenario model to execute the task.

- *Interface Objects*. Interface Objects establish two-way communication between the Intelligent Agent and the GIS. These objects receive event notifications from the GIS environment and send them to the Event Manager. Besides, these objects are used to access objects of the GIS component from the scenario model.
- *Data Access Utilities*. Databases can be accessed in two ways. The first way is to get spatial data that are used in the GIS component via the methods of the layer's class. The alternative way is to use Data Access Utilities directly. Data Access Utilities make it possible to manage spatial and non-spatial databases.
- *External Application Wrappers*. We define a wrapper as a class that contains an object to which the class provides an interface. A wrapper can wrap functions exported from a dynamic-link library (DLL), an OLE automation interface, an OLE control, a COM object, or any other set of functions. A wrapper hides implementation details of a difficult programming interface. An individual wrapper must be created for each external application. In order that a scenario program make use of services provided by an external application, an object of the appropriate wrapper class must be created in the scenario program, and the methods of this object must be invoked.

4. Implementation

This section discusses the prototype system, which implements the proposed IGIS architecture. In our experiments, we used GIS framework AKIS and created an Intelligent Agent that included the CLIPS expert system shell (Fig. 4).

CLIPS (C Language Integrated Production System) is an expert system tool that provides a complete environment for the construction of rule and/or object-based expert systems (Giarratano, 1993). CLIPS supports three programming paradigms: rule-based (knowledge is represented as heuristics), object-oriented (complex systems are modeled as modular components) and procedural (capabilities similar to those provided by C, Pascal, Lisp). When using expert systems, two kinds of integration are important: embedding CLIPS in others systems and calling external functions from CLIPS. CLIPS was designed

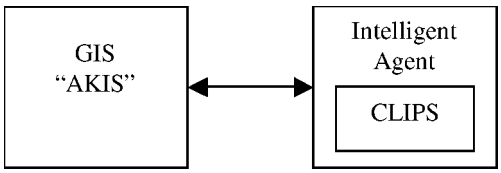

Fig. 4. Integration of the GIS AKIS and the expert system shell CLIPS.

to allow both kinds of integration. CLIPS was included into the Intelligent Agent component and extended by external functions that allowed interacting with the GIS component and external programs via interface objects of the agent and wrapper methods

AKIS is our developed object-oriented GIS framework that provides typical GIS functionality, spatial data access, and possibility to integrate it with other frameworks. AKIS can run as a stand-alone product or be extended with additional functionality. AKIS supports point, polyline, polygon, image, and grid type objects, allows viewing multiple maps simultaneously and using an overview map to see the position of map views, provides the layer manager to set layer attributes, display layer list and layer modes, provides the object geometry editor to create new objects or edit the geometry of existing objects, supports various GIS data file formats (AKIS GDB, ArcView Shapefile, DXF, DBF).

Implementation of GIS component interfaces

In the prototype system, the GIS component defines the hierarchy of objects that are accessible via their interfaces to its clients. Each object supports a Component Object Model (COM) interface for direct access to object properties, methods, and events. The intelligent agent uses these interfaces to manipulate objects in the GIS component. Fig. 5 shows the basic object hierarchy that is implemented in the AKIS system. When starting the Intelligent Agent, the GIS component passes to it the pointer of the *Application* object interface. The *Application* object is at the top, and other objects are subordinate to it. This relationship allows direct access to the objects subordinate to the *Application* object by using the properties and methods of the *Application* object.

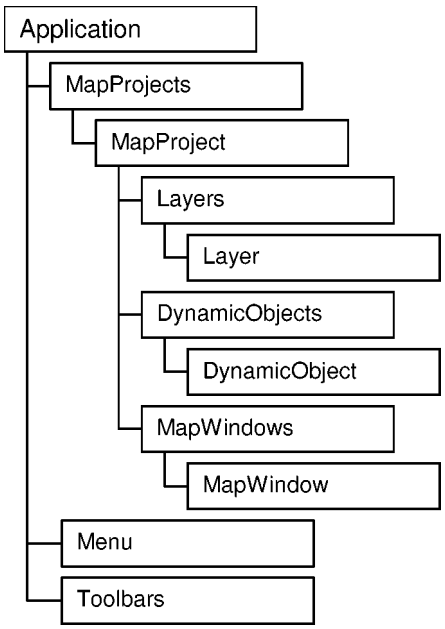

Fig. 5. GIS Component object hierarchy.

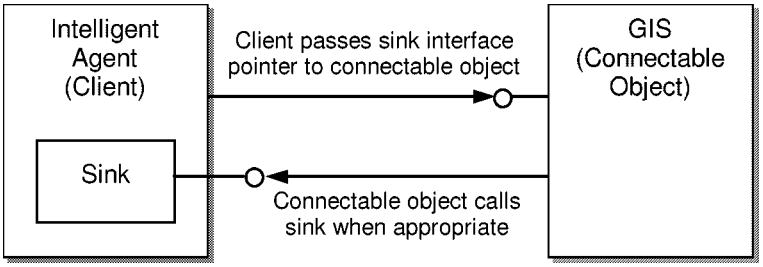

Fig. 6. The process of connecting a client sink object to a connectable source object.

Implementation of bi-directional interface

The Bi-directional interface between the GIS component and the Intelligent Agent is implemented on the basis of the connectable objects architecture (Box, 1998). A connectable object is a COM object that uses a standard COM mechanism for notifying client that something has occurred or changed. A COM connectable object provides outgoing interfaces to their clients in addition to their incoming interfaces. As a result, objects and their clients can engage in bi-directional communication. Incoming interfaces are implemented on an object and receive calls from external clients of an object while outgoing interfaces are implemented on the client's sink and receive calls from the object. The connectable object defines an interface it would like to use, and the client's object, called the sink, implements it. The basic connection process is illustrated in Fig. 6.

Implementation of the Event Manager

The Event Manager is implemented as a part of the Intelligent Agent. It accepts notifications about events from the GIS component, external devices, or scenario models. When the Event Manager is notified about some event, it creates an appropriate fact and passes it to CLIPS via its API. This fact can activate some scenario rules, which handle this event.

Implementation of wrappers

In the prototype system, wrappers are implemented as CLIPS classes. A wrapper hides implementation details of a programming interface. An individual wrapper must be created for each external application. Methods of wrappers are implemented as external CLIPS functions in C. In order that a scenario program make use of services provided by an external application, an object of the appropriate wrapper class must be created in the scenario program, and methods of this object must be invoked. The wrapper also transforms data from the CLIPS format to the external application format.

Knowledge base

The General GIS domain knowledge and scenario models are saved in separate CLIPS files. The general knowledge describes general constructs, classes, and rules that can be used in scenario models. For example, descriptions of classes *Geometry*, *Geoobject* are general knowledge. A scenario model includes specific domain knowledge needed to solve a concrete task.

The described prototype system was implemented in C++ and runs under Windows 98. Real applications were created on the basis of the prototype system: AKIS-R – system for radio waves propagation analysis (Paliulionis, 1996), AKIS-AVL – automatic vehicle location system (Paliulionis, 2000).

5. Conclusions

The presented intelligent GIS architecture enables us to dynamically extend GIS functionality by using an intelligent agent. It allows us to use domain-specific knowledge and reasoning mechanisms. The intelligent agent monitors events, which occur in the GIS environment, and executes tasks depending on user's actions. The proposed architecture can be effectively implemented by using an expert system shell and COM connectable objects mechanism that enables the implementation of the bi-directional interface between the client (the intelligent agent) and the server (the GIS component).

References

Andrienko, G., and N. Andrienko (1999). Making a GIS Intelligent: CommonGIS Project View. *AGILE'99 Conference*, Rome, pp. 19–24.

Box, D. (1998). lit Essential COM, Addision Wesley.

Buehler, K., and L. McKee (1998). *The OpenGIS Guide. Introduction to Interoperable Geoprocessing and the OpenGIS Specification.* OGC Technical Committee of the Open GIS Consortium, Wayland (MA).

Fedra, K. (1993). Expert systems in water resources simulation and optimization. Stochastic hydrology and its use. In J.B. Marco *et al.* (Eds.) *Water Resources Systems Simulation and Optimization*. Kluwer Academic Publishers, The Netherlands. pp. 397–412.

Fedra, K. (1995). Decision support for natural resources management: models, GIS and expert systems. *AI Applications*, **9**(3), 3–19.

Fitzgerald, R.W., and B.G. Lees (1996). Temporal context in floristic classification, *Computers and Geosciences*, **22**(9), 981–994.

Foody, G.M. (1997). Fully fuzzy supervised classification of land cover from remotely sensed imagery with an artificial neural network, *Neural Computing & Applications*, **5**(4), 238–247.

Giarratano, J.C. (1993). *CLIPS User's Guide*. NASA/Lyndon B. Johnson Space Center Information Systems Directorate, Software Technology Branch.

Ishikawa, Y., T. Furudate, S. Uemura (1997). A wrapping architecture for IR systems to mediate external structured document sources. In R. Topor and K. Tanaka (Eds.), *Database Systems for Advanced Applications'97*, Vol. 6 of Advanced Database Research and Development Series. World Scientific Publishing Co., Singapore. pp. 431–440.

Jacobson, I., M. Ericsson, A. Jacobson (1994). *The Object Advantage – Business Process Reengineering with Object Technology*. ACM Press.

Miller, D.M., E.J Kaminsky, S. Rana (1995). Neural network classification of remote-sensing data. *Computers & Geosciences*, **21**(3), 377–386.

Openshaw, S., and I. Turton (1994). *Building New Spatial Interactive Models Using Genetic Programming*. School of Geography, University of Leeds, Leeds, UK.

Paliulionis, V. (1996). GIS naudojimas radijo bangų sklidimo analizei. In *Lietuvos mokslas ir pramonė. Informacinės technologijos-96.* Konferencijos pranešimų medžiaga, Kaunas, pp. 166–171 (in Lithuanian).

Paliulionis, V. (2000). Architecture of an Intelligent GIS. Databases & Information Systems. *Proceedings of the 4th IEEE International Baltic Workshop*, Vol. 2. Vilnius, Technika. pp. 193–198.

Papakonstantinou, Y., A. Gupta, H. Garcia-Molina, J. Ullman (1995). A query translation scheme for rapid implementation of wrappers, *Lecture Notes in Computer Science*, Vol. 1013, pp. 161–186.

Rodrigues, A., J. Raper, M. Capitao (1995). Development of spatial intelligent agents. In *Proceedings of the First International Conference on Spatial Multimedia and Virtual Reality*. Museum of Water, Lisbon, Portugal, pp. 18–20.

Sandhu, R., and P. Treleaven (1996). Client-server approaches to model integration within GIS. In *Third International Conference/Workshop on Integrating GIS and Environmental Modeling*, Santa Fe, New Mexico, USA.

Wu, X.P., and Y.X. Zhou (1993). Reserve estimation using neural network techniques. *Computers & Geosciences*, **19**(4), 567–575.

V. Paliulionis is a PhD student in computer science at the Institute of Mathematics and Informatics. In 1990 he graduated from Vilnius University. His current research interests are geographic information systems.

Intelektualizuota GIS: architektūra ir realizavimo metodai

Viktoras PALIULIONIS

Pateikiama intelektualizuotos GIS architektūra, kuri įgalina išplėsti GIS galimybes naudojant dalykines žinias ir samprotavimų mechanizmus. Šioje architektūroje intelektualizuotas agentas stebi įvykius, vykstančius GIS aplinkoje, ir priklausomai nuo vartotojo veiksmų atlieka įvairias užduotis. Intelektualizuotas agentas turi ekspertinės sistemos branduolį ir žinių bazę. Naudojamas hibridinis žinių vaizdavimo būdas, apimantis produkcijų taisykles, klases, objektus ir funkcijas. Aprašyta prototipinė sistema, sukurta pagal siūlomą architektūrą.