An Approach to Hypermap-Based Applications

P.S.C. Alencar, D.D. Cowan, T.R. Grove and M.A.V. Nelson
Computer Systems Group, Department of Computer Science
University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada
Fax: (519) 746-5422
E-mail: {alencar,dcowan,trg,guta}@csg.uwaterloo.ca

C.I. Mayfield
Biology Department
University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada
Fax: (519) 746-0614
E-mail: mayfield@sciborg.uwaterloo.ca

Abstract

Recent advances in interactive user interfaces and database technology are leading to significant changes in the role of maps in GIS systems; maps are becoming the starting point for both spatial analysis and data recovery processes rather than the final product. In this paper we present an architecture for map-centered (or hypermap) applications that reflects this shift in emphasis. The hypermap provides a direct interactive interface to geographic, numerical, textual and multimedia data. Our proposed architecture currently uses a tool called the "Map Engine", and the architecture is flexible enough to allow rapid development of new hypermap-based applications. The architectural structure and the tool have been validated in several real-world applications, two of which are outlined in the paper. The eventual goal of this research is to generate application-specific software systems by combining this architecture with the results of a dialogue with the domain expert. This paper reports on the architecture and associated tools and shows how applications are currently structured.

Keywords

Hypermap, multimedia components, architecture, browser

1 INTRODUCTION

Recent advances in software technology, including interactive user interfaces and databases, are leading to a significant change in the role of maps in GIS environments (Maguire, Goodchild and Rhind 1991). Maps are now being seen

Environmental Software Systems Vol.2 R. Denzer, D.A. Swayne & G. Schimak (Eds.)

as a way to query databases dynamically as well as to present information. This new way of representing and interacting with spatial objects requires a different architectural approach to building contemporary GIS systems. Interaction with the spatial objects in the interface is only one key consideration. Since the data and tools in many cases have already been developed and predate the new form of GIS system by several years, we need to be able to integrate legacy data and tools.

In application domains where spatial location is a key component, maps can be used to link, order, and display geo-referenced information. What experts in these application domains use are no longer plain flat maps, but hypermaps (Laurini and Thompson 1992, Kraak 1996), where the geo-coordinates are used as a starting point for the retrieval of data about a specific map object or a collection of map objects. The user of the hypermap chooses an object or group of objects, and then is directed to the type of data available. Available data varies over a wide range, and can include numbers, text, and many types of multimedia components such as sound, images, and animations. Thus, the individual data elements are linked to maps to present the semantics related to the objects in the map, and to enhance the geographic information for further exploration, analysis, and presentation.

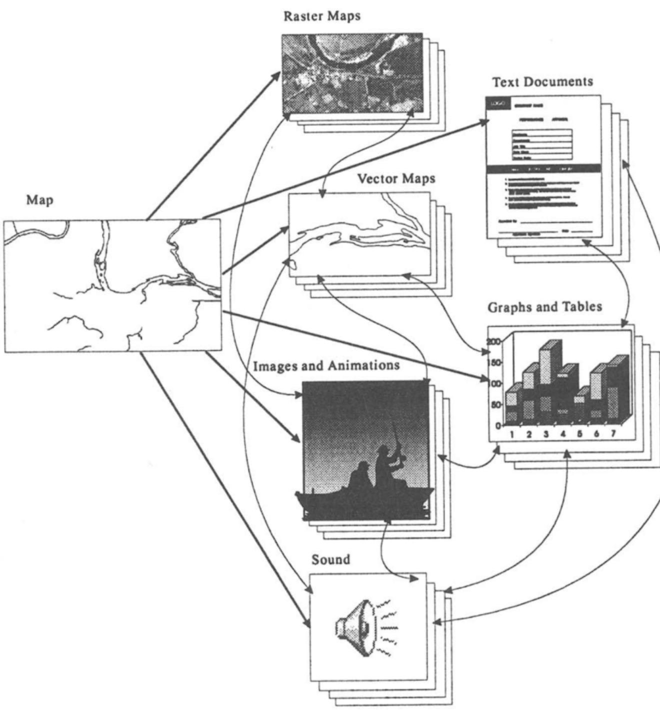

Figure 1 An example of a Hypermap Based Application

An example illustrating a hypermap-based application is presented in Figure 1, where a map of a river network is the starting point for information access. Selecting a specific river on the map can give the user access to more detailed data such as: other maps in vector or raster format; photographs of the region; text documents describing water quality; numeric data on water flow and pollution indices; tables and graphs showing river water levels; video images showing sports or activities on the river; and audio clips about the preservation of the river.

In this paper we present an approach to creating hypermap-based applications and illustrate the associated architecture. Such an architecture would not only assist programmers and domain experts, but might also enhance the ability of the non-computer expert to build an application in a specific problem domain. We also describe the "Map Engine" a tool used to present the spatial database and to support the interface with the spatial objects on the screen.

The approach has been validated on several real-world examples, which provides a strong indication that our architecture is flexible enough to support a broad class of hypermap-based applications. Two of these examples are described in this paper.

2 A MODEL FOR HYPERMAP-BASED APPLICATIONS

Our eventual goal is to assist the domain expert in creating hypermap-based applications using the underlying architecture and the information supplied through a dialogue between the computer system and the expert. A system architecture is the first step in understanding this process of generating hypermap applications.

Our architecture allows the integration with maps of various types of semantic information such as numeric data, text, sound, images, tables, graphs, and animations. The map is drawn using a primary browser called the "Map Engine" that extracts a vector representation of the map from a spatial database, while secondary browsers are used to display data associated with the objects in the map. References to the secondary browsers and their interfaces are stored in a table. In addition, each browser has a feature table describing how to display the semantic information for a requested object. Figure 2 illustrates this architecture.

Each object in the spatial database is identified by an M,L,O key which determines the map, layer and object identifiers. Selecting an object retrieves its M,L,O key and determines the associated secondary browsers. The system then calls an interface program with some parameters that are obtained in the feature table associated with each browser. The feature that has the same M,L,O identification as the object selected in the map is passed to the interface program, and is used to tell the external browser how to display the semantic information. For example, consider a textual description of a house. In the

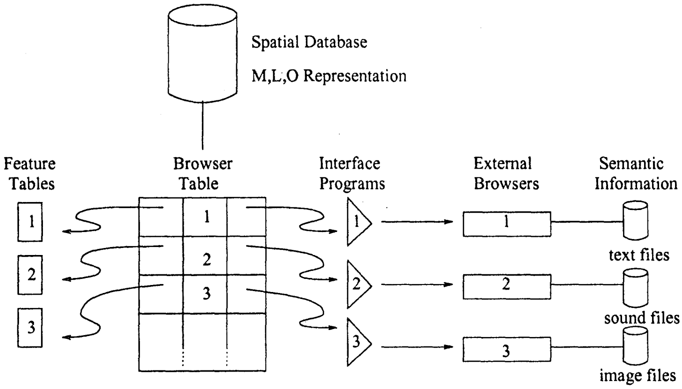

Figure 2 Map Engine's Architecture

feature table we store the hypertext keys to tell the hypertext browser how to display the text. When the external browser is dispatched, the semantic information is retrieved from a text database.

Each available browser is represented by one line in the browser table and has one feature table and one interface program associated with it. Browsers can include viewers, analysis and modeling tools such as: spreadsheets (Microsoft Excel or Lotus 1,2,3), hypertext browsers (Inforium LivePAGE (The Information Atrium Inc. 1995)), database managers (Sybase SQL Anywhere, Oracle), and bitmap viewers (Adobe Photoshop).

In this architecture, some tables are fixed and have to exist, such as the map table, the browser table, and feature tables, while some store the semantic data for each application. After deciding on the semantic portion of the schema we must complete the tables converting data from the sources provided. Some information has to be entered manually, for example, the data that establishes the relationship between the objects in the map and their semantic information. The browser table and the feature tables have to be completed later.

We have two cases to consider when we build a new application. In the first one, the type of the information is already available to the schema and so the appropriate interface program to call the browser is referenced. In this situation we need only to add entries to the associated feature table. In the second case, the type of semantic information is unknown to the schema and a interface program needs to be developed to allow the new browser to be incorporated into the system. In addition, a new feature table needs to be created.

The Map Engine was designed and developed to be used as a component in more complex systems. In general, each application that can be developed using the Map Engine as a component has general operations (such as display the map or select objects in the map) and very specific operations (such as fill a polygon in a map or find the distance between two points in a map).

The general operations can be composed using the resources provided by the Map Engine, and the specific operations can be implemented using the Map Engine's interface for communication between the parts of the application.

The architecture of the hypermap system is open in that new map objects and secondary browsers can be added with relative ease. Although entering the semantic data may be tedious, we do not have to change the design, in the worst case we may have to add a database table. We use a client-server architecture supported entirely by an SQL database, where the client and server can be geographically separated and connected by a network.

3 TWO APPLICATIONS USING THE MAP ENGINE

Land development in certain regions of Canada must be preceded by a heritage study of the region to ensure that important information about its people, culture, values, and environment will not disappear. The Blair application was created for this purpose, and is composed of raster and vector maps, photographs of houses and places, and text documents about each object in the map.

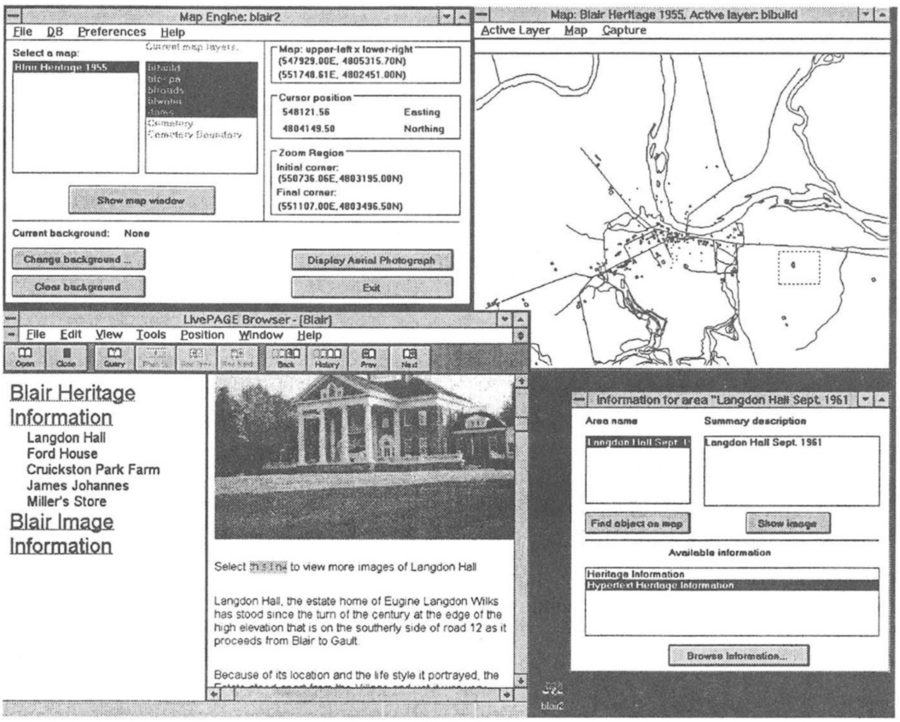

Figure 3 Blair Application

A sample screen from the Blair application is shown in Figure 3. The first window in the top left corner of the Figure shows the database information: map layers selected to be displayed, the extent of the region, the cursor position that shows the geographic coordinate of the current position, and the geographic extent of the zoom region. In this window we also have a button for an aerial photograph of the Blair region which can be used as background for the map of Blair. In the right side we have the map window showing the map of Blair. The map contains 4 different layers: buildings, roads, water, and dams which are displayed in different colors on the screen. The user can select an object, and then obtain its semantic information. The third window (clockwise from top left) then shows the name of the object and the different browsers to display the semantic information The window in the lower left-corner shows a hypermedia browser (LivePAGE) displaying information about the selected object.

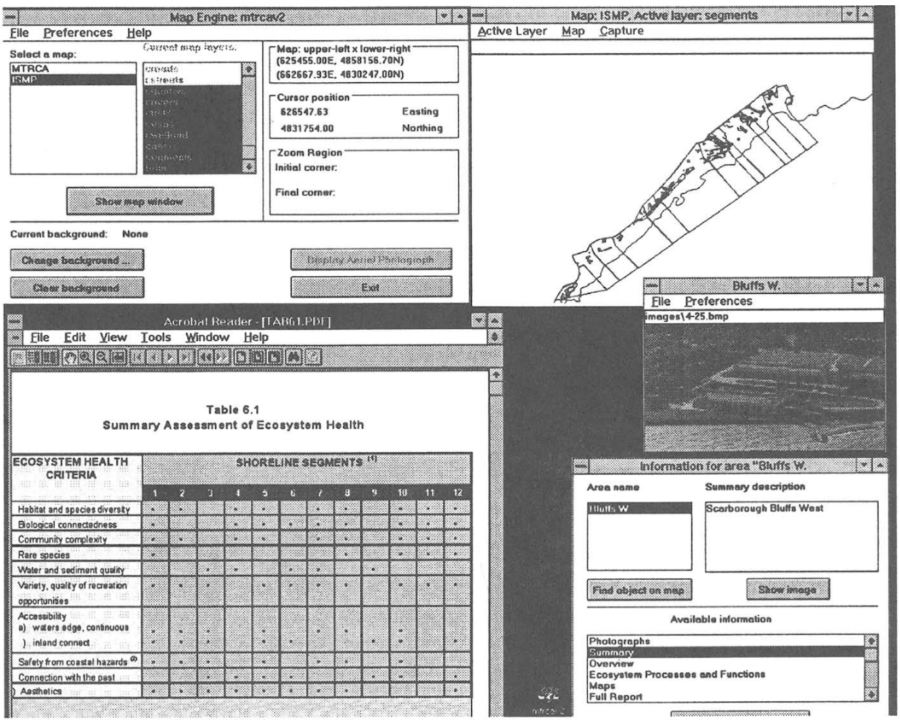

Figure 4 Toronto's Shoreline Application

The Integrated Shoreline Management Plan (ISMP) application was developed to demonstrate what is happening to the shore of Lake Ontario in the Greater Toronto Area (GTA) as the ISMP is taking effect, and a sample screen is shown in Figure 4. The first window again shows information about the database and its layers. The map window shows the shoreline divided into 12 segments. By selecting a segment the user can see all information related to that segment. We see in the figure a photograph showing a building on the lakeshore and a table summarizing what you can find in each segment. Both were displayed using different browsers. The table was displayed using Adobe Acrobat and the photograph by a bitmap viewer.

Other applications that incorporate modeling and analysis tools are under development.

4 FUTURE WORK AND CONCLUSIONS

In this paper we present a model for building hypermap-based applications based on a general open architecture and tool called the "Map Engine". We describe a manual approach for the domain expert or programmer to construct different types of hypermap applications using this architecture. The Map Engine and architecture have been used to develop new applications quite quickly. In general our experience indicates that the amount of time needed to customize the system for new users or new applications is acceptable. The Blair application, for instance, was built in less than two days.

Even though applications can be constructed quickly, extensive programming expertise and database knowledge are required from the domain expert. Our ultimate goal is to develop a generator that conducts a dialogue with the user while instantiating a hypermap-based application based on the system architecture and existing tools. Thus, we need to create a formal description and framework to express the relationship of the components and the interfaces in a specific application.

5 ACKNOWLEDGMENT

This work has been supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), the Brazilian Federal Agency for Graduate Education (CAPES), and Sybase (Waterloo).

REFERENCES

Kraak, M.-J. (1996). Integrating Multimedia in Geographical Information Systems, *IEEE Multimedia* pp. 59–65.

Laurini, R. and Thompson, D. (1992). *Fundamentals of Spatial Information Systems*, Academic Press.

Maguire, D. J., Goodchild, M. and Rhind, D. (1991). *Geographical Information Systems*, Longman.

The Information Atrium Inc. (1995). LivePAGE Browser: Tutorial and Reference Manual, *Technical report*, Inforium, Waterloo, Ontario, Canada.

6 BIOGRAPHY

P. S. C. Alencar received the Ph.D. degree in computer Science from the Catholic University of Rio de Janeiro (PUC-Rio), Brazil, in 1992. He is a Research Assistant Professor in the Department of Computer Science at the University of Waterloo, Waterloo, Ontario, Canada. His research interests are in the areas of software engineering and formal methods in formal engineering. Particularly, he is interested in software specification and design methods, processes, tools, and reuse; object-oriented systems specification, design, and development; design patterns and frameworks; viewpoints; software architecture; software validation and verification; and formal design models. He is a member of the ACM, IEEE, CIPS, ACFAS, and AAAI.

D. D. Cowan is Professor Emeritus of Computer Science at the University of Waterloo and Director of the Computer Systems Group, and has been at the University since 1960. He received a B.A.Sc. in Engineering Physics from the University of Toronto in 1960 and an M.Sc. and Ph.D. in Applied Mathematics from the University of Waterloo in 1961 and 1965 respectively. Professor Cowan is the founding Chairman of the Computer Science Department at the University of Waterloo. His research interests include software engineering, highly interactive software systems and the application of computers to education and environmental management.

T. R. Grove is a staff researcher in the Computer Systems Group at the University of Waterloo. He works primarily in the area of programming language design and implementation. He has lectured extensively on programming techniques to professional development audiences.

C. I. Mayfield is a Professor of Biology at the University of Waterloo, heads the Environmental Informations System Project there and is an Assistant Director of the United Nations University - International Network for Water, Environment and Health (UNU-INWEH) where he is responsible for Information Technology and Systems. His research is centred around the areas of groundwater microbiology, biodegradation and bioremediation of contaminants in soils and groundwater ecosystems, data sharing methods and environmental information systems.

M. A. V. Nelson is a PhD student in the Department of Computer Science at the University of Waterloo. In 1995 she received a MSc in Computer Science at the Federal University of Minas Gerais, Brazil. Her research interests are object-oriented analysis and design, frameworks, and software architectures applied to the geographical and environmental systems.