WEB BASED DATA VISUALIZATION INTERFACE FOR SPATIAL DATABASES

Siniša Mastelić Ivić, Hrvoje Matijević, Miodrag Roić
Institute of Engineering Geodesy and Spatial Information Management,
University of Zagreb - Faculty of Geodesy
Email: ivic@geof.hr, hmatijev@geof.hr, mroic@geof.hr

Abstract: Today a great deal of accumulated spatial data fund resides within spatial databases (SDBMS). Some of it is 2D, some is 3D and finally there is also a certain amount of it enriched with a temporal component. Unfortunately, unlike traditional CAD or GIS based spatial data management systems, spatial databases do not come with "off the shelf" graphical user interface for spatial data visualization. Querying is naturally possible in alphanumerical form. All sorts of CAD and GIS systems being able to retrieve and generally manage data within SDBMS can of course be used for those purposes, but with two important drawbacks: they cost additional money and must be installed on client's computer. Then, there is a variety of "map servers" but also burdened with costs and limited to two-dimensional displaying. This situation can be overcome, and surprisingly enough, with less effort. With only a medium level of programming knowledge one should be able to prepare his own database query visualization interface, leaving it either low-level to be able to directly write, execute and visualize his data access queries or coding them into the interface to make it accessible to users with less or no SQL experience. Using Servlet/JSP, a powerful server side scripting technology based on Java programming language and VRML as the standard for 3D spatial data visualization on the Web, fully Web based and user platform independent interface is developed. Server side application written in Java receives user requests, fetches the data from the database, does all the necessary conversions and preparations and serves it to the user in a standard VRML format. The user interface for displaying and manipulation of spatial data stored in SDBMS, modeled in up to three dimensions and requiring only VRML viewer to be installed on a client computer, is described in this paper.

1 Introduction

Visualization of two dimensional spatial data is fully developed and widely used today. On the other hand 3D data, or 2D data with for instance temporal component have been limited to CAD level management systems, or Web based fixed content visualization solutions. Users of building level (Facility Management) to nation-wide (Cadastral) information systems require vendor independently and efficiently accessible vivid spatial data.

2 Technologies used

Three technologies are used to design core of the described data access and visualization engine, namely VRML (Virtual Reality Markup Language) for visualization, Servlet/JSP (Java Server Pages) [4] in the middle tier programming logic and JDBC (Java Data Base

Connectivity) for communication with the database. Although increasingly being replaced with X3D as a richer and more flexible technology, VRML is still present in general area of both static and dynamic 3D data visualization. Vast amount of existing worlds in the first place and wide range of available viewing and authoring tools in the second are main factors for that. Should the main drawback of using VRML as opposed to X3D, which is its inability to manage descriptive data, not be generally over restrictive, one can use it just as well, as shall be presented later in text. VRML and Java can be efficiently combined using VRML's EAI (External Authoring Interface) through which elements of the scene graph can be manipulated in either existential (creation, elimination) or degenerative way (modification).

In the past, several server side scripting languages have shown their worth (PHP, Perl, ...) as used to dynamically generate web page content. Today two major technologies have pushed those back, namely Microsoft's ASP (Active Server Pages) and JSP (Java Server Pages) developed by Sun Microsystems. Generally those are quite similar, with two main exceptions. Pages built using JSP technology are typically implemented using a translation phase that is performed once, the first time the page is called. The page is compiled into a Java Servlet class and remains in server memory, so subsequent calls to the page have very fast response time whereas in ASP the page is recompiled for every request. JSP can be seen as a more convenient way of writing servlets. Secondly, JSP is designed to be both platform and server independent. In contrast, ASP is purely a Microsoft based technology deployed primarily on Windows based servers, and designated for Microsoft's IIS web server.

JDBC technology is an API for Java (included in both J2SE and J2EE releases) that provides cross-DBMS connectivity to a wide range of SQL databases and access to other tabular data sources, such as spreadsheets or flat files. This enables the middle tier logic to seamlessly access different data sources and combine the data. Some alternative technological platforms for such an interface are described in [7].

3 The Interface

The general idea of multi-tier information system architectures is the separation of data on one side and user (interface) on the other, through deployment of extensible business logic in the middle. The data access interface consists of visualization engine and data access and preparation engine. Benefits from such an approach are firstly evident in data security and consistency increase and secondly overall (especially large scale) system maintenance simplification.

The middle tier of a information system, developed in Java programming language and implemented as a JSP application (servlet), is the data access and preparation engine of the interface. Visualization engine is a standard Web browser with VRML plug-in. The general usage scenario initiates with user accessing designated Web page, formulating a query either using prepared query definition engine or formulating it from scratch. The application (servlet) sends the query via JDBC to the database and retrieves the data. This is a general usage scenario for any three tier database access system and is trivial to implement. Conversion of retrieved spatial data, stored in database native or any other format [6], is then performed by the servlet. Our implementation uses CyberX3D Java classes [2] for the conversion. Next, spatial data set is, over TCP/IP, served to the user in VRML format thus the only necessary software required on the client machine besides Web browser is a VRML plug-in. This level of interface development is quite generic and can be quickly implemented because query pre-processing is left to the user. For SQL experienced users especially in data and system model development phases this can pose a significant help (Figure 1).

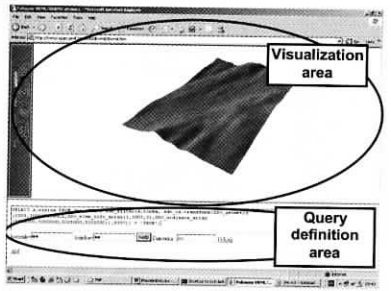

Figure 1: Basic data access interface layout

Limitation of VRML in descriptive data manipulation can be overcome by adding a sub-interface for retrieval and displaying of additional (descriptive) data (Figure 2).

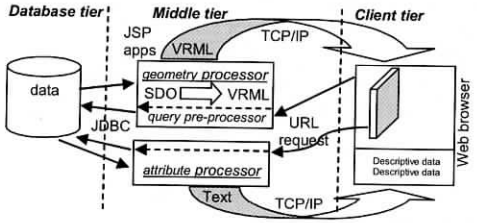

Figure 2: Enhanced data access interface architecture

Furthermore, using VRML EAI it is possible to implement additional functionality for data manipulation and send it to the user via a applet. For instance a simple applet could change the transparency of selected objects using predefined criteria selection. In our implementation Oracle10g database is used for data storage with Oracle's own JSP/Servlet container. Although this is a simplifying circumstance, installing independent JSP/Servlet containers, like Tomcat [1], shouldn't prove significantly more complicated. The client tier of the interface can be fully developed to enable simple process from query definition to the displayed data manipulation (Figure 3).

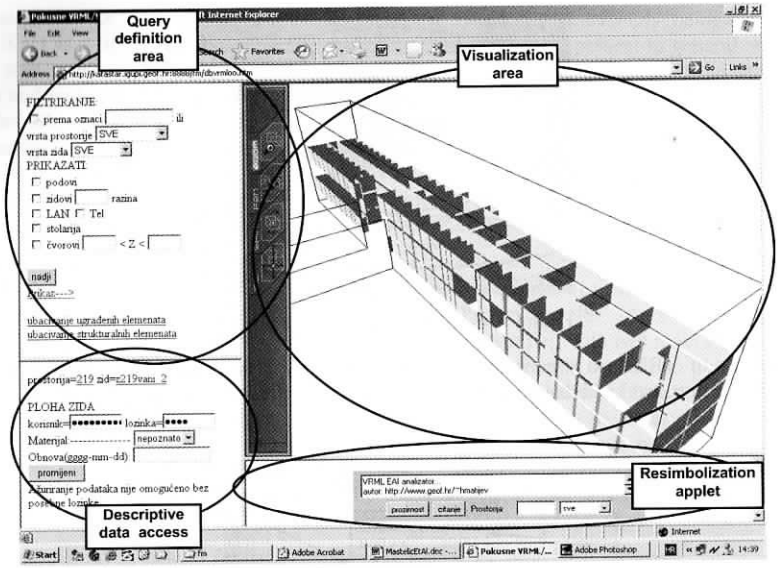

Figure 3: Fully developed data access interface layout

3.1 The servlet, data engine

The core component of the interface is data retrieval and conversion engine (geometry processor). After query pre-processor prepares the query according to user needs, it is posted to the database via JDBC and data is received in JGeometry (Java form of SDO_GEOMETRY) and a standard Java format for descriptive data. A VRML scene graph is then initialized and prepared for the addition of nodes. The main program sequentially passes JGeometry objects together with selected limited set of descriptive data (generally object ID or type) to the conversion subroutine. Upon the recognition of the JGeometry object type (point, curve, polygon) the subroutine adds the VRML node accordingly (Sphere, IndexedLineSet, IndexedFaceSet) to the scene graph. VRML's Anchor node is "wrapped" around geometry nodes to add limited set of descriptive data and a URL of a second servlet (attribute processor) used for more extensive attribute data retrieval. Additional benefit from VRML's geometric data enrichment using at least some attribute data comes with deployment of a simple EAI data manipulation applet.

3.2 The applet, visualization engine enhancement

Once the data is fetched, converted and served to the user it becomes static in the view of analytical capabilities. Further analysis and data manipulation can of course be performed by re-querying with the cost of additionally loading the server hardware, and with at least one order of magnitude slower response time. Provided that the fetched data set is enriched with some limited (VRML doesn't allow extensible) attribute data, EAI can be used to manipulate it. The general idea is to use EAI in the applet to read the entire scene graph, analyze geometric nodes and extract the added attribute data. Once a handle to each scene graph's

node is acquired, and its attributes known the entire scene graph or any subset of it can be redrawn with colour, transparency and (or) other attributes changed.

4 Usage cases

Besides 3D city models like [3] which are often researched and described topic of 3D visualization, several other areas of its usage exist.

Facility management systems are generally 3D data oriented and development of its data model [5] should be followed by appropriate visualization interface. Be it a power plant, airport or a business building, visualization of its data can and should be in all three dimensions. Providing that system's data storage is a SDBMS, 3D visualization interface can increase its overall performance in view of user's requirements and data interpretation. It being Web based, furthermore boosts data availability which is also important.

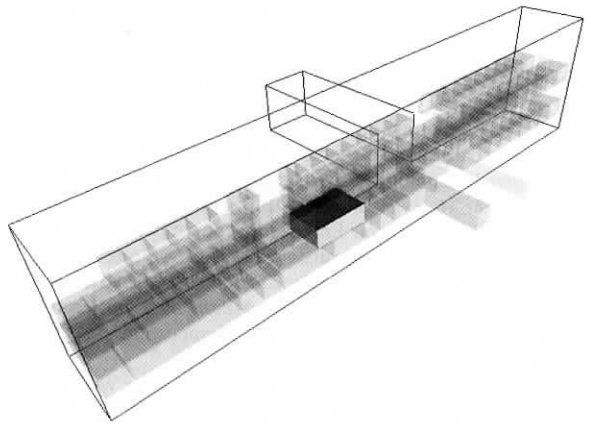

Figure 4: Transparency used for revealing distinct part of the building

Previously described FM system is only the most obvious case where 3D visualization is necessary or can be used. Two dimensional spatial data today often includes a temporal component. A simple and efficient way of representing this quasi third dimension can also be achieved using the described interface (Figure 4).

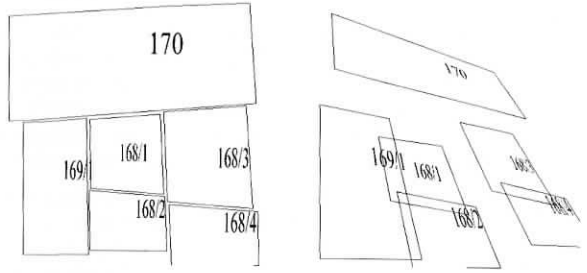

Figure 5: Cadastral parcels with temporal component as 3-rd dimension

Using time as the third dimension of the data one can visualize previous states of any selected area or changes which caused it.

5 Conclusions and further research

The described data access and visualization interface has proven its capabilities and flexibility. Once a spatial data management system's data is modeled, acquired and stored in a SDBMS the interface can be used for access, based on only standard Web access components (browsers and plug-ins). Java as a programming platform proves as most adequate because of its simplicity, stability and cross platform availability. VRML, although slowly falling behind X3D is still capable of satisfying nearly all levels of users needs.

In the future X3D and its XML origin should prove even more flexible especially in the area of distributed GIS where more attribute data should be transferred and processed on client side.

References:

[1] Apache Jakarta Tomcat, http://jakarta.apache.org/tomcat/index.html
[2] CyberX3D, http://www.cybergarage.org/
[3] Gruen, A., Wang, X.: Urban data management with a hybrid 3D GIS, Proceedings of UDMS'99: 21st urban data management symposium, Venice, 1999.
[4] J2EE Java Servlet Technology, http://java.sun.com/products/servlet/index.jsp
[5] Matijević, H., Roić, M., Ivić, S. M.: SDBMS Based Data Model for CAFM Systems, 3rd International Conference on Engineering Surveying and FIG regional Conference for Central and Eastern Europe, Bratislava, 2004.
[6] Oosterom, P. van, Stoter, J., Quak, W., Zlatanova, S.: The Balance Between Geometry and Topology, In: Dianne Richardson and Peter van Oosterom (eds.); Advances in Spatial Data Handling, 10th International Symposium on Spatial Data Handling, Springer-Verlag, Berlin, 2002.
[7] Vries, M. de, Stoter, J.: Accessing a 3D geo-DBMS using Web technology, ISPRS Joint Workshop on "Spatial, Temporal and Multi-Dimensional Data Modelling and Analysis", Québec, Canada, 2003.