Web Service Based Modular Architecture for 3D Web Visualization of Geo-referenced Data

Igor Antolović[1], Miroslav Milivojević[2], Dejan Rančić[3], Vladan Mihajlović[4]

Abstract – **This paper presents a service based architecture for 3D Web visualization of geo-referenced data which relies on the GinisVis library. The GinisVis visualization library combines server .NET and client AJAX/WebGL technology as well as modular workflow approach in order to efficiently visualize geo-referenced data obtained from geospatial Web services like WFS and WMS. As a proof of concept a prototype Web application for 3D visualization of terrain and geo-referenced objects is created. It is shown that this architecture presents a flexible and stable platform for future rapid development of 3D Web GIS applications.**

Keywords – ***Modular, 3D Web visualization, Web service***

I. Introduction

Web service based architectures are becoming increasingly popular in the past few years. Applications of existing Web services are various ranging from language translation, word search, weather information, 2D geo-referenced map rendering etc. Communication with Web services is accomplished using open standards such as SOAP (Simple Object Access Protocol) which defines the XML data message format as well as WSDL (Web Services Description Language) which was designed with the aim to provide Web service description.

Availability and standardization of Web services has greatly influenced the migration of existing desktop architecture based applications into the Web environment.

Existing solutions for client-server based 3D data visualization rely mostly on thin Web clients. These are simple Web clients which are able to display a 3D model image which is entirely generated on the server side. This approach has shown to be very reliable in the past since Web technologies did not allow 3D rendering on the client side. Other existing alternatives require installation of additional plug-ins which would lead to many problems in terms of compatibility and usability.

In this paper, special attention will be devoted to Web applications for 3D data visualization which have become practically possible with the recently introduced WebGL standard [1] which provides hardware support for OpenGL ES 2.0 within the standard Web browsers.

The aim of this paper is to consider a Web service architecture, which should serve as a solid and flexible platform for efficient development of 3D Web GIS (Geographic Information Systems) applications [2,3] that would enable geo-referenced 3D object rendering using open standard Web technologies. It is important to emphasize that the starting point in designing this architecture is based on a workflow methodology. The basic concept of this idea is shown in Fig. 1.

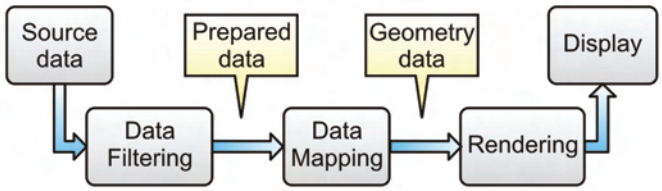


Fig. 1. Basic workflow graph

The data from the source domain is first filtered and then mapped on geometric primitives which can be rendered and than displayed. Within this kind of system the data can pass through one or more filters. Each filter usually implements a relatively simple function which processes the input data by additionally taking into account various input parameters.

This paper first gives an overview of existing 3D Web data visualization solutions followed by a description of the GinisVis client-server architecture. Finally a minimum set of modules is introduced in order to enable modeling of a 3D terrain as well as 3D geo-referenced buildings.

II. Related Work

There are many examples of desktop environments that provide a user interface for interactive workflow creation in order to obtain rather complex data visualizations. Among such applications one example is VisTrails[4] which is a scientific workflow management system that provides rich support for data exploration and visualization and relies on the VTK[5] (Visualization Toolkit) library that implements a wide set of algorithms for 3D data processing. This and similar environments have influenced the development of various Web based data visualization applications in the past several years. Among existing 3D Web visualization solutions ParaviewWeb[6] is distinguished by its robustness and capabilities. This environment is based on a thin client which provides 3D data display as well as an intuitive user interface, while the rendering is done on the server side. Paraview also relies on VTK thus offering a wide set of data visualization

[1]Igor Antolović is with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Nis, Serbia, E-mail: igor.antolovic@elfak.ni.ac.rs.

[2]Miroslav Milivojević is with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Nis, Serbia, E-mail: miroslav.milivojevic@elfak.ni.ac.rs.

[3]Dejan Rančić is with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Nis, Serbia, E-mail: dejan.rancic@elfak.ni.ac.rs.

[4]Vladan Mihajlović is with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Nis, Serbia, E-mail: vladan.mihajlovic@elfak.ni.ac.rs.

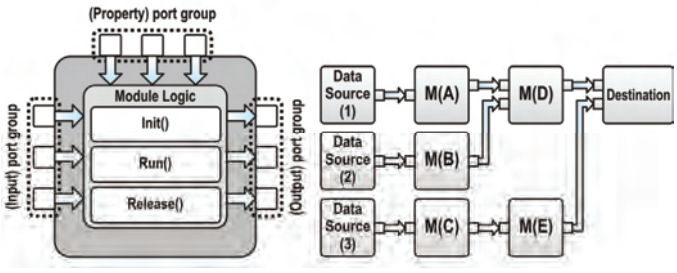
algorithms which can be somewhat observed as a kind of a downside of this and similar environments. Namely the user must choose between great number of algorithms in order to get the best output results thus making his task sometimes time-consuming. Another solution which is designed to work in a client-server environment and deals with the previously described problem is Envision[7]. What is distinctive about this solution is that it provides the user the ability to describe the basic features of the input data hence enabling the system to offer only a reduced set of visualization algorithms. In this way, the user is greatly relieved because he does not have to choose manually between hundreds of available algorithms. Unlike previous solutions, [8] describes a Web service that is based on the IRIS Explorer system. The described architecture relies on a Web service which generates VRML (Virtual Reality Modeling Language) models hence the client side needs a VRML viewer application that can be installed as a browser plugin.

What is common for most of this solutions is that they rely on thin clients therefore providing the highest level of compatibility with existing Web browsers. However, as WebGL API is becoming the leading standard for 3D Web visualization on the Web it is now possible to create true Web 3D applications capable of fully using modern graphic acceleration hardware.

The combined concepts of workflow based data visualization and WebGL enabled clients directly influenced the design of the GinisVis Web service architecture.

III. GINISVIS ARCHITECTURE

The GinisVis Web client-server architecture is built on top of the server side GinisVis.NET as well as the client side GinisVis AJAX library. Both of this libraries are in fact a migration of the GinisVis C++ library described in [9].



a) b)

Fig. 2. a) Single module architecture, b) Basic workflow example

The GinisVis library supports a workflow methodology which is based on assigning parts of the visualization process to smaller functional units in the form of modules. In the process of visualization, every module should execute a relatively simple processing of the input data and generate an appropriate output. A simplified architecture of a single module is shown on Fig.2.a).

The real advantage of this modular approach lies in the possibility of module linking as shown in Fig.2.b). It is important to note that the corresponding output ports of module M(A) and M(B) must be compatible with the corresponding input ports of module M(D). Also in the process of execution of modules, M(D) can be executed only after executing the modules M(A) and M(B). After execution off all modules a final destination output is created. More precisely the described architecture is designed to generate outputs in a form of 3D models.

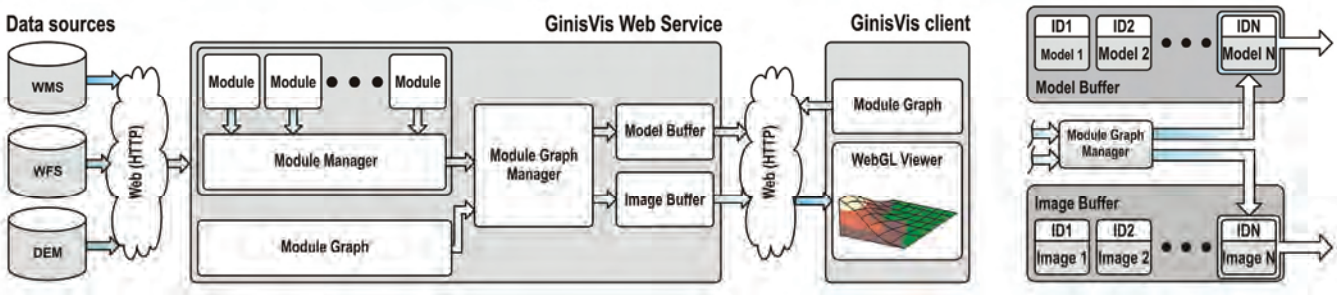
A detailed view of the client-server architecture is shown in Fig.3.a), where we clearly distinguish three parts:

- **Data sources** – they can be various but standard OGC (*Open Geospatial Consortium*) Web services are preferred like WMS (*Web Map Server*) which provides geo-maps for requested regions and WFS (*Web Feature Server*) which provides geo-feature information,
- **GinisVis Web Service** – relies on the GinisVis.NET library, contains a set of module implementations and descriptions and provides an engine for module workflow creation and execution,
- **GinisVis Client** – relies on the GinisVis AJAX library and provides rich 3D rendering of models retrieved from the service.

The GinisVis Web service consists of the following components:

- **Modules –** a set of DLL libraries and corresponding XML descriptions where each module can implement data filter, data source or an algorithm,
- **Module Manager –** is a component that performs registration of all available modules,
- **Module Graph –** is a structure that represents the current data flow graph that is executed,
- **Module Graph Manager –** is a component that performs workflow graph execution.

The client retrieves generated 3D models in a two step procedure. The first step is to obtain only the geometry of the model that contains texture URLs, resulting in an additional texture download request. These problems can be solved by introducing Image Buffer and Buffer model components as shown in Fig.3.b). The Image buffer is a component that provides temporary storage of textures on the server side allowing them to download through sequential call to the GinisVis Web service. More precisely, this component assigns each image a unique ID which is used to link with the corresponding 3D model. After the texture is retrieved it is been automatically deleted from the buffer. On the other hand the Model Buffer component has an identical role as the Image Buffer but it handles geometry instead. This two components form an efficient buffering mechanism which overcomes the inability to send both the model and all textures at once.



a) b)

Fig. 3. a) GinisVis client-server arhitecture, b) GinisVis buffering subsystem

Considering all service side components the communication between the GinisVis client and the GinisVis Web service can be described as follows:

1. The client invokes the *GetCapabilities* service side method.
2. The Web service responds with a list of available modules.
3. The client invokes the *RunModuleGraph* Web method sending a XML workflow configuration of linked modules.
4. The service receives the request, initializes dynamic module loading, performs module linking, executes the workflow and sends back the generated XML output in a form of a 3D model list.
5. The client retrieves the generated output and starts model downloading by invoking the *GetModel* Web method. After model retrieving the client invokes subsequent *GetImage* requests in order to obtain model textures.

In order to test the concept of the described architecture, necessary modules for terrain and 3D building modeling are considered below. As already stated, our primary motivation of is to design a platform that will provide efficient future development of 3D Web GIS applications.

IV. VISUALIZATION

Visualization of 3D terrain models has always been the first and foremost requirement in GIS applications. This resulted in a great number of algorithms (both static and dynamic [11]) for 3D terrain visualization. However, it is not our goal to create a fully optimized algorithm for terrain visualization but rather to transpose the process of terrain modeling into a workflow graph.

Basically a terrain model consists of a height matrix based mesh overlayed with a geo-referenced texture for a given region. The starting point for generating such a mesh is the data flow graph shown on Fig.4.a) (bottom).

This data flow graph consists of the following modules:

- **WMS proxy** – a proxy module that retrieves a geo-referenced image from a WMS service.
- **DEM (Digital Elevation Map) proxy** – a proxy module that retrieves a height matrix from a DEM service.
- **Terrain Modeler** – a module that generates a simple 3D triangle geometry based on a input height matrix.
- **Mesh 3D** – a module that combines both texture and geometry and produces a textured 3D.
- **Project** – a module that converts geo-spatial coordinates into Cartesian coordinates.
- **ColladaExporter** – a module that exports 3D Mesh into XML COLLADA [10] models.

In contrast to terrain models, 3D buildings can have a moderately complex structure which can make the process of their modeling a rather difficult task, but one of the most practical ways of modeling is based on contour elevation as shown on Fig.4.a) (top).

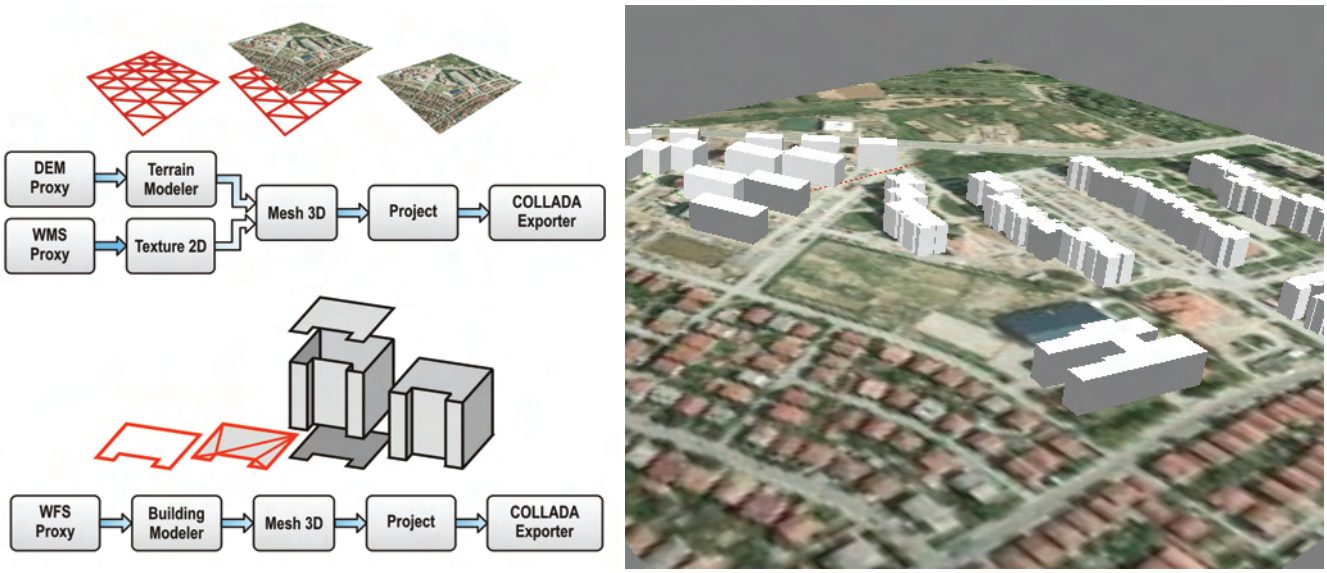
Some of the modules required for building modeling are reused but also additional modules were created:

- **WFS proxy** – a proxy module retrieves a collection of features from a WFS service by invoking a *GetFeature* request. What is important is that every feature consists of great number of attributes among which the *Geometry* attribute represents a 2D geo-referenced contour.
- **Building modeler** – a module that generates a 3D building model based on a input contour geometry obtained from WFS features.

In both cases the **Project** module has a very important role since it converts the model from geo-referenced space into Cartesian space. The input models can have geometries which are positioned in one of the standard geo-coordinate systems (e.g. WGS84 where every point is determined by a longitude, latitude and height value) while the output model has standard Cartesian (x,y,z) coordinates. There are two types of models which can be handled by this module:

- **Multipoint geo-referenced –** these are the models that are geo-referenced for every point in their geometry. The terrain for example is one such model.
- **Singlepoint geo-referenced –** these are the models that are geo-referenced with a single point usually the center of mass. In our case 3D buildings belong to this category.

The final result in a form of a combined 3D model of a terrain and buildings is shown on Fig.4.b) where the rendering is performed using the WebGL based GinisVis AJAX client which enables interactive visualization of COLLADA models.



a) b)

Fig.4.a) Terrain and building workflows b) Final result rendered using the GinisVis AJAX WebGL based client

V. Conclusion

The lack of proper support for 3D visualization within the Web browsers has resulted in Web data visualization solutions that rely mainly on thin clients which are able to show images of 3D models that are fully rendered on the server side. The recently introduced WebGL standard that provides support for OpenGL ES 2.0 enables hardware rendering within Web browsers. This technology enables the design of more powerful Web based 3D model rendering clients eliminating the need to install additional browser plug-ins.

In this paper, we introduced the GinisVis Web client-service architecture that takes advantage of both current Web open-source standards and workflow graph methodology. The idea of a workflow approach is that the Web service handles a collection of modules which individually perform rather simple functions but connected within a workflow graph they can produce rather complex 3D models. Also, the GinisVis client server architecture emphasizes the advantage of Web technologies and the usage of standard OGC services as data sources (WMS, WFS, etc.) as these services play an important role in GIS applications.

A proof of concept was provided by creating a minimum set of modules that allow 3D terrain and geo-referenced buildings modeling by using a WMS service as a source of aerial photographs and WFS service as a data source for building contours.

The main advantage of the GinisVis Web service is that it forces open standard protocols as well as technologies, and furthermore, it is based on a modular workflow architecture thus it can be easily extended with new modules. All this features make this architecture an ideal platform for building future rich 3D Web GIS applications.

References

[1] http://www.khronos.org/webgl/
[2] D. Rančić, D. Dačić, “Ginis web 3D modeler - a framework for 3D terrain visualisation on web”, 8th AGILE Conference on GIScience, May 26-28, 2005, Estoril Congress Center, Estoril, Portugal
[3] D. Rančić, A. Dimitrijević, V. Mihajlović, "GIS and Virtual Reality Systems Integration", ICEST 2004, Bitola, Macedonia, pp. 313-316, 2004.
[4] Steven P. Callahan , Juliana Freire , Emanuele Santos , Carlos E. Scheidegger , Cláudio T. Silva , Huy T. Vo, „VisTrails: visualization meets data management“, Proceedings of the 2006 ACM SIGMOD international conference on Management of data, June 27-29, 2006, Chicago, IL, USA
[5] William J. Schroeder , Kenneth M. Martin , William E. Lorensen, „The design and implementation of an object-oriented toolkit for 3D graphics and visualization“, Proceedings of the 7th conference on Visualization '96, p.93-ff., October 28-29, 1996, San Francisco, California, United States
[6] K.M. Martin, B. Geveci, J. Ahrens, C. Law, „Large Scale Data Visualization Using Parallel Data Streaming “. IEEE Computer Graphics & Applications, (July 2001)
[7] G. P. Johnson, S. Mock, B. Westing, G. S. Johnson, EnVision: „A Web-Based Tool for Scientific Visualization“, CCGRID '09 Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid IEEE Computer Society, Washington, DC, USA, 2009
[8] J. Wood, K. Brodlie, J. Seo, D. Duke, „A Web Service Arhitecture for Visualization“. ESCIENCE '08 Proceedings of the 2008 Fourth IEEE International Conference on eScience IEEE Computer Society, Washington, DC, USA, 2008
[9] I. Antolović, V. Mihajlović, D. Rančić, M. Milivojević, “GinisVIS: Data flow control graph based 3D visualization framework”, YUINFO 2010, Kopaonik
[10] Dejan Rancic, Aleksandar Dimitrijevic, Bratislav Predic, "Spatial Coherency and Parallelism in Blocks Reorganization of RINGO Algorithm for Large Terrain Rendering", WSEAS Transaction on Computers, Vol. 5, No. 12, pp. 3073-3079, December 2006,
[11] http://www.khronos.org/collada/