Using COLLADA and X3D for WebGL Based 3D Data Visualization

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

Abstract **- This paper describes the evaluation of *COLLADA* and *X3D* with the GinisVis *WebGL* based library for *3D Web* data visualization. These standard formats are very similar but they are quite different in their design goals and intended use. This paper describes a universal parser that was developed with a goal to provide uniform interface for both formats. The results of this study are comparisons of COLLADA and X3D parser performance in the form of tables.**

Keywords **– COLLADA, X3D, WebGL, 3D Web, visualization**

I. Introduction

WebGL[1] based 3D data visualization is the goal of a large number of organizations that develop advanced 3D Web applications. This is a substitute for desktop environments for 3D visualization[2,3]. The 3D Web applications provide a visualization of complex scenes, whose rendering is accelerated with graphics hardware. These have a major impact on research and development of Web visualization in the scientific fields such as Computer graphics, Astronomy, Physics, Chemistry, Medicine and etc. WebGL[1] is derived from OpenGL ES 2.0 and provides similar rendering functionality. It extends the capability of the JS (JavaScript) programming language to allow it to generate interactive 3D Web applications. This technology uses the HTML5 canvas element which is accessed using DOM (Document Object Model). A large number of functionality and 3D scene description is contained in portable formats such as COLLADA[4,5], VRML and X3D[5,6].

This paper describes a universal parser that was developed with a goal to provide uniform interface for both formats such as COLLADA and X3D. This parser is part of the GinisVis Ajax library[7], together allow a 3D interactive Web visualization as well as full compatibility with the well known Web browsers such as Firefox 4.0, Chrome 11, Opera 11 and Safari 5.

The goal of this paper is to present a set of capabilities of these two formats, their purpose, as well as similarities and differences. The possibilities of these formats are great and some of them allow users of CAD (Computer aided design), GIS (Geographic Information Systems)[8] and automation applications to enjoy the benefits of open standard royalty free content formats. The COLLADA and X3D section describes their feature set is expanding to incorporate technologies such as packaging programmable shader[9] effects and controlling real-time physics engines.

Next section describes the parsers architecture, as well as their common interface and strategy to the selection one of them. The results of this study are 3D data visualization in a Web browsers as well as values comparisons of both formats in tabular form. The conclusion section describes the analysis of the both formats, their comparison, the intersection features as well as their purpose values based of research results.

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

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

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

II. COLLADA and X3D

COLLADA and X3D are two open standards that use XML schema technology to fully describe 3D scenes. These formats can be used together as a powerful tool set for developing interactive Web and enterprise applications.

A. COLLADA format

COLLADA (Collaborative Design Activity) defines an interchange file format to enable interactive 3D applications. This format is developed by the non-profit consortium, the Khronos Group[10]. COLLADA is an intermediate format that is focused on describing 3D models and bringing content and assets from different authoring tools to an 3D application or an another tool. This is useful for DCC (Digital Content Creation) tools, conditioning tools and for data storage. Rich data representation that fully describes one or more 3D scenes is stored in files with the ***.dae*** extension. This XML format contains a large number of elements that on mathematical or physical way describe the scene models as well as their interaction. Some of these elements are: **geometry**, **node**, **material**, **effect**, **image**, **scene**, **dynamic, controller, density**, **mass**, **inertia**, **imager** (image sensor), **bind, texcombiner** (texture combiner) **etc.** Mathematical model and design model are presented with the first five elements respectively. Element **geometry** contains the position coordinates, texture coordinates, colours and normals. Element **node** can contain the geometry and the other nodes. The COLLADA file can contain one **scene** that is composed of tree nodes and cameras. Element **dynamic** contains a Boolean value that indicates whether a physical model is movable or not. Element **controller** is a generic mechanism to describe active or dynamic content and it contains an element that describes control data. This is a very important element that manages the operations of another object with a control data. The following elements such as mass and density describe the model more clearly. Total mass of the physical model is described in the floating-point value of the **mass** element. Diagonal elements of the moments of inertia are presented with three floating-point numbers of the **inertia** element. The **imager** element is an image sensor of camera and defines how camera sensor transforms light colour and intensities into numerical values. The **bind** element allows the mapping of parameters to uniform inputs at run time. Parameters such as material, texture, and etc. can be forwarded to a shader allowing the FX (framework) runtime but they can also be the shader programs. The **texcombiner** element allows the specification of different commands for combiner-mode texturing. Extra tags allow definition of user information and provide additional information about or related to its parent element. The **instance_node** element refers to an element in the node tree or in another COLLADA file using the **url** attribute. This is especially convenient for GIS[8] applications for example when the city visualization requires that each object or building is stored in one COLLADA file.

B.X3D format

X3D is the successor to the VRML (Virtual Reality Modeling Language) standard. This format fully describes the 3D scene, physics, interactions and collisions between objects on the scene. Described 3D data are stored in files with ***.x3d*** and ***.x3dv*** (X3D VRML) extensions. Because the file is encoded in XML it can be easily extended thus making X3D more flexible than its predecessors. Developers use other technologies such as DOM and XPath for X3D data access. Unlike other formats X3D describes 3D models in a very simple way even without the uses of specification. This is not a binary format and not designed as an intermediary format.

X3D is a delivery format that contains the information needed for interactive applications and includes behaviors such as picking, viewing, navigation and scripting. This format is used in many applications and it is applied in the following areas: virtual worlds, social networks, GIS, military simulation, medical visualization, industry engineering (oil and gas, automotive, virtual training), mobiles and etc. X3D data can be created from DTD (Document Type Definition) or XML schemas and transformed using XSLT (Extensible Stylesheet Language Transformations). There are a large numbers of elements in X3D format such as geometry, grouping, appearance, material, textures, viewing, environment, navigation, animation, lightning, sound, interaction and etc. This format contains a 2D geometry that includes following elements **arcs**, **closed**, **circles**, **polylines**, **rectangles**, **triangles** and as 3D geometry includes **cones**, **cylinders**, **elevation grids**, **extrusion**, **boxes**, **spheres**, **indexed face sets** and **indexed line sets**. Defining following elements such as terrain, water and skies is provided by the **GeoElevationGrid** element**.** Information of physical characteristic description of the viewer's avatar and models are stored in the navigation info element. Full 3D navigation of X3D format is supported with the following types: walk, any, examine, look-at, none, slide, pan and fly. 3D environment experience in the X3D scene is enabled by the following elements such as lightning, sound, background and fog element. For example describing the earth and sky is allowed by the background element with a set of colors or image textures. Sound and audio clip element allow the identification and management of sound in the X3D scene. X3D supports following compressed and uncompressed formats such as wavefile (***.wav***), MIDI (***.midi***) and MP3 (***.mp3***). Element **inline** embeds an X3D scene into another scene using the **url** attribute. Url refers to another X3D file that contains data for the scene such as a list of children nodes, prototypes etc. This element is important for the development of GIS applications. Behaviors in the run-time X3D environment is allowed by the events. Some events are initiated by the time sensor and for other events the following sensors are responsible: touch sensor, key sensor, picking sensor, visibility sensor, plane sensor, sphere sensor, geo touch sensor and so on. Main purpose of sensors is to control the animations and user interactions. For example when the user clicks or drags a geometry followed by a series of activities. First the touch or plane sensor is activated, then it activates the time sensor which sends values to an interpolator node and interpolation of result allows the modification of current geometry in the scene.

III. Parser Architecture

This section describes the similarities and differences between COLLADA and X3D formats, the design goals and the communication between objects of developed classes.

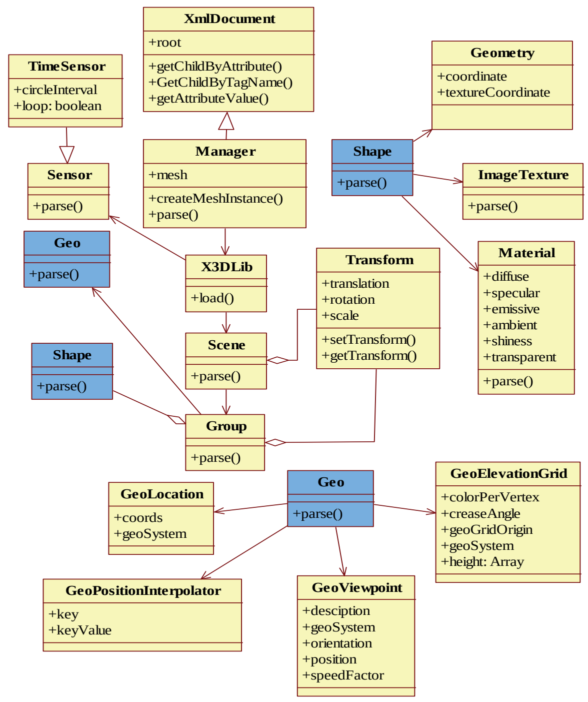

Fig.1. X3D parser architecture

A library that allows you to select one of X3D and COLLADA parsers is developed in this study. Decision on the choice of a parser is made using the file extension.

XmlDocument class is the basic XML parser. This parser consists of methods to search and return value of the attribute node, then methods for searching nodes and return of the node based on their tag name, as well as methods for searching nodes and return of the node based on attribute name or based on attribute values.

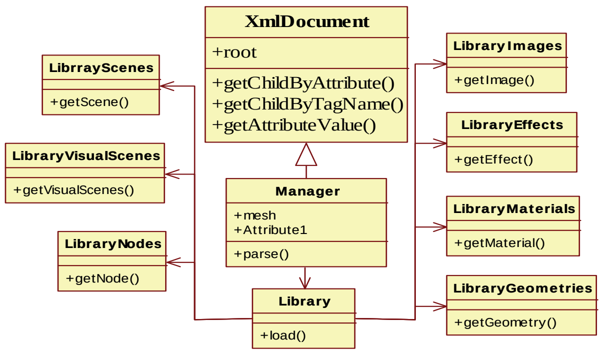

Fig.2. COLLADA parser architecture

Manager class initiates the main parse method. This method runs load method of the X3Dlib class. These classes are common for both COLLADA and X3D parsers.

Scene class contains all the scene elements covered in the X3D format such as group, shape, viewpoint and so on. Unlike the COLLADA parser where there is an class (LibraryVisualScenes) which refers to the instance of several scenes in the file X3D parser contains only one Scene instance, because each X3D file can only include one X3D scene according to XML principles.

Group class of the X3D parser is very similar to the LibraryNodes class of COLLADA parser but it's easier for the parsing. LibraryNodes class contains a list of nodes that one node has a geometry and material or refers to another node in this class. Group class contains a tree of nested nodes and consequently reduces the parsing time or the time for access to nodes. Nodes in Group class can refer to other group instances or may have an instance of Shape class.

Shape class is similar to the node that contains geometry and doesn't contain other nodes. This class contains a reference to Geometry class and Material class.

Transform class allows the transformation of the Group object, such as translation, rotation and scaling.

Geometry class contains information about vertices, indices, normal and texture coordinates. Unlike COLLADA parser where the material is related with geometry using material ID attribute, Geometry and Material are embedded in the Shape class and X3D parser doesn't contain attributes to connect them. Common features of both formats are to include the field indices of points, normals and texture coordinates. Otherwise, the graphics hardware for visualization uses OpenGL ES 2.0 that supports a unique index field common to all types of coordinates. The parsers allow the creation of a unified field of all indices.

GeoViewpoint class provides a perspective view of the scene using a geospatial data. This class contains several attributes such as **geoSystem**, **geoOrigin**, **position, orientation**, **navType** and **speedFactor**. The geoSystem attribute contains a combination of the following fields of strings that define which ellipsoid is used and how. The position attribute defines the actual coordinate of the viewpoint. The speedFactor attribute is a multiplier to the elevation-based velocity.

GeoElevationGrid class presents a uniform grid of elevation values and GeoLocation class allows the georeferencing of object whose is parent node.

GeoPositionInterpolator class allows interpolation of geographical coordinates using the key values and information about the specified coordinate system. Classes that allow the description of georeferenced objects are integral part of the X3D parser because only X3D format has specialized nodes for this purpose.

TimeSensor class represents a timer in the scene and allows the construction of events based on time. Object of this class provides information that can be used for many purposes and some of them are: using it for simulation and animation, use it as a trigger for periodic events, use it as an as alarm clock and so on. In this case the TimeSensor class is used for animation of georeferenced objects.

IV. Results

This paper describes the realized sub-library for 3D data visualization based on the WebGL technology using COLLADA and X3D standards. This sub-library is an universal parser that allows the extraction of 3D content from the described formats. The universal parser consists of the COLLADA and X3D parser where both are advanced XML parsers. Parser classes are implemented using the COLLADA and X3D formats as well as their methods that successfully provide the necessary information such as geometry, material, normal and colors for 3D visualization to the Web browser. In addition, support for georefrenced objects is implemented. Parsers are implemented using the JavaScript API. There are several limitations of the used technology and some of them are: OpenGL | ES2.0 and WebGL do not support 32-bit values for the vertex indices. WebGL supports identifier names of no more than 256 characters. The maximum number of vertices per chunk is 64K. 3D data visualization is provided on several best-known Web browsers. Rendering of complex 3D scenes is executed on the graphics hardware without having to install additional plug-in components. Result of the 3D data Web visualization is an imported 3D model from the COLLADA file, shown in Fig.3. The values shown in Table 1 are the results of comparing parsing time and download time. Testing is performed with the same 3D models for the COLLADA and X3D parser.

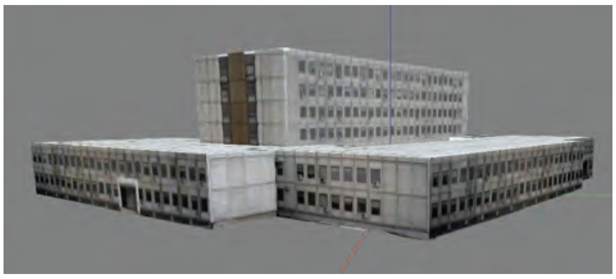

Fig.3. Health Center[11]

Table 1 COMPARISON OF FORMATS

Format	COLLADA		X3D		COLLADA and X3D		
Model/Parameters	download time(s)	parsing time(ms)	download time(s)	parsing time(ms)	number of faces	number of vertices	framerate (fps)
Health Center	3	81	1.38	67	80	40	6.5
Hotel Ambasador	4	186	1.85	155	1628	1648	6.5
Court	2	376	0.92	336	2586	1712	6.5
Niški cvet	11	2381	5.08	2198	7954	11757	4.25
Federation	32	2060	14.77	1895	1357	550	7
Post Office	18	2185	8.31	1908	1096	525	7.5
Mileševa	20	20528	9.23	20100	4804	3223	7
Colosseum	4	1637	1.85	1520	9988	5104	7

V. CONCLUSION

COLLADA and X3D are two very similar XML based open standards. Their goals, design and purpose are very different but both are used in tools for data visualization.

COLLADA is an intermediate and interchange file format. This format is used to describe 3D models in the following fields: computer graphics, visual simulation, animation, virtual reality, Medicine, GIS and so on. COLLADA was designed for the interchange applications and for transporting data from one tool to another including: **Maya, 3D Studio Max, Sketchup, Blender** etc. COLLADA file can contain more than one scene and can contain only one **scene_instance** tag within **scene** tag. This information is stored in the element as libraryVisualScenes. In COLLADA file, nodes have a reference to other nodes or contain rich information about geometries and materials. Materials effect allows the use of color or texture or both. This format contains sensors and one of them is the imager that describes the camera sensor characteristics. Described 3D content of the COLLDA does not support the ECMA script which would allow the interactive features.

X3D is a delivery format and open standard based on XML schema technology. This is a good format for interactive 3D Web applications. X3D supports shaders and the following techniques such as **CSM** (Cascade Shadow Maps), **SSAO** (Screen Space Ambient Occlusion), as well as reflection and lightning in the real-time. X3D can contain only one scene nodes and the tree nodes. Nodes are embedded in each other so that it reduces the time while moving through the tree nodes. X3D format supports many sensors that allow animation, simulation and interaction in the 3D scene. Scripting is another good feature of this format that is allowed by the X3DScriptNode element. This allows the browser to execute the program in the Script node's url field. The Script node's url field allows the following scripts: Java, ECMA Script and inline ECMA Script. There are many purposes of this format and some of them are for the following fields: CAD, Chemistry markup language, GIS, Networking, Humanoid Animation, Medical and etc.

COLLADA and X3D can be used for developing powerful tools or 3D Web applications using modular Web services[12]. Both formats allow the insertion of user information and import of the other COLLADA and X3D files. This is very important for GIS applications because it is possible to combine several files of these formats and provide a textual description of the 3D scenes.

REFERENCES

[1] WebGL, Khronos Group: http://www.khronos.org/

[2] Igor Antolović, Vladan Mihajlović, Dejan Rančić, Miroslav Milivojević, "GINISVIS: DATAFLOW CONTROL GRAPH BASED 3D VISUALIZATION FRAMEWORK", YU INFO 2010, Kopaonik, pp. 1-6, March 2010, Proceedings

[3] Igor Antolović, Miroslav Milivojević, Dejan Rančić, Vladan Mihajlović, "3D Object modeling and Visuallization using Modular Web services", YUINFO 2011, Kopaonik

[4] COLLADA: http://www.khronos.org/files/collada_spec_1_4.pdf

[5] COLLADA, X3D: http://www.khronos.org/collada/presentations/Developing_Web_Applications_with_COLLADA_and_X3D.pdf

[6] Craig Anslow , Stuart Marshall , James Noble , Robert Biddle, "Evaluating X3D for use in software visualization", Proceedings of the 2006 ACM symposium on Software visualization, September 04-05, 2006, Brighton, United Kingdom

[7] Miroslav Milivojević, Igor Antolović, Dejan Rančić, "AJAX LIBRARY FOR THE VISUALIZATION OF 3D DATA ON THE INTERNET USING WEBGL TECHNOLOGY", YUINFO 2011, Kopaonik

[8] D. Rančić, A. Dimitrijević, V. Mihajlović, "GIS and Virtual Reality Systems Integration", ICEST 2004, Bitola, Macedonia, pp. 313-316, 2004

[9] Shader: http://www.opengl.org/documentation/glsl

[10] Khronos Group: http://www.khronos.org/

[11] Health Center: http://sketchup.google.com/3dwarehouse/details?mid=56bdbc35d9a2e3f2369024cdcd1746d&prevstart=0

[12] Igor Antolović, Miroslav Milivojević, Dejan Rančić, Vladan Mihajlović, Marko Kovačević, "3D terrain models generating using modular web service", 18. Telekomunikacioni forum TELFOR 2010, Sava Centar Beograd, pp. 1-4, November 2010, Proceedings on CD