Large Scale Network Topology Visualization System Based on Three.JS

Yuxiao Wang, Yuanzhang Li, Yu'an Tan, Quanxin Zhang* and Jun Zheng
School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China
Beijing Engineering Research Center of Massive Language Information Processing and Cloud Computing Application, Beijing 100081, China
*Corresponding author

***Abstract*—with the rapid development of visualization, network visualization has become an important branch, which is applied in many aspects of life such as business, education and virtual community. But existed network visualization systems are based on small volume and static effect. Along with the incessant expansion of modern internet scale, traditional tools' practicality decrease. Showing huge information of network clearly and effectively will be of great signification. In our paper, large-scale network topology visualization system is finished by using Web 3D technique—Three.JS in process of building and analysis, which can display topology clearly. As middleware, Node.JS can transfer communication data forward or backward instantaneously, which can solve the problem of dynamic and real-time 3D effects. Node-Webkit framework helps realize its cross-platform feature. This solution can run smoothly and clearly under the 10^3 order of magnitudes network topology. It makes large scale network visualization more direct and convenient.**

Keywords-web 3D; three.JS; network topology; visualization; node.JS

I. INTRODUCTION

Visualization technology which can translate complex data into geometric shapes or images and organizes meaningful information in multidimensional spatial form will help users obtain and understand data details and real-time change explicitly [1].

With the increase of network coverage, network size and function also have been extended. How to show network relationship and information in a reasonable way has become an important problem. Network topology visualization technology has developed into an important branch [2]. Currently, there are some organizations which have a lot of research progress, e.g. CAIDA, NLANR. Many visualization tools have been realized to expand visualization tool sets. Meanwhile, their methods and results provide reference to the next research.

Through JAVA 3D API, Walrus [3] of CAIDA can generate simple or complicated, three-dimensional object which can be deformed with color, transparency effects. Walrus network nodes will use three-dimensional distribution of divergent tree display. At the same time JAVA 3D is implemented in the JVM level, making development less difficult. However, JAVA 3D released by Sun has not been widely used. Software development technology doesn't develop and update as expected. Coupled with the vigorous development of other 3D technologies, JAVA 3D has gradually been abandoned. Therefore, as for now, JAVA 3D it is not a good solution. Walrus static nodes can only read static data, and cannot provide dynamic update feature in the meanwhile

Cichild tool is very good network data visualization system of National Laboratory of Applied Network Research (NLNAR) [4]. Its dynamic raw data is usually in the form of a bar graph or node / edge displayed. Cichild was designed to study the network performance including TCP or wireless network, and there's no particular attention or analysis for changes by real-time data and network attacks or failures. Cichild is written in C programming language, using the open source OpenGL library mesa, under Windows95 and many UNIX platform compilers [5]. The cross-platform capability is poor and does not have general applicability for which it's not a good visualization system tools.

The chosen solution in this system is based on Web GL technology which is a new trend of Web 3D and can display three-dimensional scenes and objects by the system graphics card and support GPU acceleration. Its performance and efficiency is particularly impressive. Web GL open source engine Three.JS [6] is in a stage of rapid development. Not only the use of Three.JS improves the speed of development, but also the technology innovation is also being able to respond to problems encountered in the development process. The combination of Web GL and HTML5 creates complex and beautiful navigation and system display is more intuitive. The use of Node-Webkit framework can pack network visualization system into the desktop applications. It crossed the limits of software systems and the environment, achieved a visualization system Cross platform. On the other hand, as middleware, changing data transmission of Node.JS will be the real-time dynamic visual display in the visualization system. These are some advantages of our system that current network visualization solutions and existing tools are not able to achieve or perfect.

II. BACKGROUND

A. HTML5 and CSS3

HTML5 adds element Canvas which is equivalent to a canvas in new edition. Using Canvas API, you can draw a variety of graphics and images including 3D images. HTML5

also provides zoom functions or other functions related to user interaction.

Cascading Style Sheets controlling page display can realize the page content separation. CSS3 in new edition has some new feature, such as gradient transparency effects, animation effect. So HTML5 and CSS3 are chosen to realize the design visualization system by adding menu bar, navigation bar, tips etc.

B. *Three.JS*

Web GL which is a 3D drawing standard, can combine JavaScript and OpenGL ES 2.0.After that, Web GL can provide 3D accelerated rendering for Canvass and display 3D models smoothly in browser by virtue of graphics card. Meanwhile, with the help HTML5 and JavaScript, it can create clean interface and data visualization [7].

Three.JS is a third-party JavaScript library for Web GL. Simple and systematic API encapsulates original Web GL. Three.JS provide a lot of 3D functions, which can create camera, light, material and objects [8]. JavaScript's computing power is increased greatly by v8 engine. The 3D effect based on JavaScript has its own advantage no matter in visual effect or in performance, which enhances user experience.

Using Three.JS reduces work load compared with other technology. It has lots of superiority such as open-source, approachable usage and high efficiency.

C. *Node.JS*

Node.JS – also called Node, is JavaScript environment. It's based on Google's runtime implementation – the aptly named "v8" engine. V8 and Node are mostly implemented in C and C++, focusing on performance and low memory consumption. Unlike in most other modern environments, a Node process doesn't rely on multithreading to support concurrent execution of business logic. It's based on an asynchronous I/O eventing model. Meanwhile, Node.JS supports the eventing module at the language level.

III. NODE-WEBKIT FRAMEWORK ANALYSIS

Node-Webkit combines Node.JS and Webkit technology. As a client application which can be used in Linux, Windows, Mac OSX platform, it encapsulates the chromium browser kernel module and Node.JS. Node.JS can use and control DOM to operate local resources.

A. *Node-Webkit Basic Structure*

The basic structure of Node-Webkit is shown in Figure 1. Package.json in the root of directory is program's configuration file which can define the start page, the name, the window appearance and characteristics. Index.html is the application's start page. The directory of JS/CSS/Resources stores JavaScript files, HTML files and images to ensure the application's styles. The file directory of node_module stores Node.JS extension components. When Node-Webkit launches the application, first of all, package.json will be loaded and basic properties will be initialized. And then additional steps will be performed.

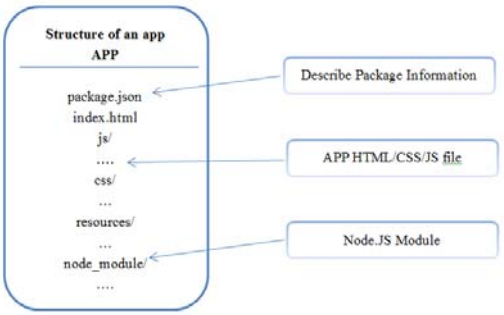

FIGURE I. INTERNAL STRUCTURE

B. *Node-Webkit Basic Functions*

Node-Webkit supports JavaScript, HTML5, CSS3 and other web technologies. It provides native API which controls application's interfaces display to access code. Node.JS provides a variety of third-party modules and native API interface, which can finish the function to access local files, access internet, and request server. The specific functions will be described in Figure II.

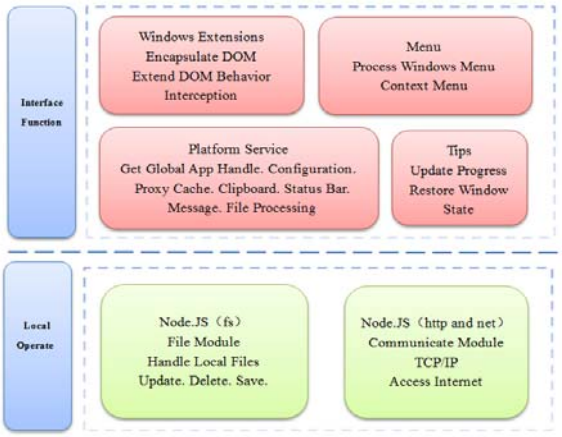

FIGURE II. NODE-WEBKIT FUNCTIONS

IV. IMPLEMENT

A. *Process of Building 3D Scene*

Three.JS renders to draw 3D object and graphics in canvas. However, the 3D space in canvas is not real. It mathematically simulates 3D world on a 2D screen. So a 3D coordinate system (x-axis, y-axis, and z-axis) to be reference can make 3D effect more straightforward and real.

We need three basic components, scene, camera and renderer to build a 3D model by Three.JS. The scene is a container for all objects. After you put objects into scene, objects have chance to be displayed. The whole process will be shown below. First of all, you need load index.html. And you would get canvas's handle. Then you can use Three.JS to realize scene function, camera function, light function and renderer function. You need to draw Sphere Geometry and Line to be Nodes and Links by Three.JS. At last, after all objects have been added into scene, they will be shown in the screen.

B. *Network Topology Layout Anaysis*

In this visualization system, nodes are distinguished into different level using different layout. Nodes which are in different level are in different layer. This method can separate different nodes and corresponding network connection. In different level, we adopt different layout algorithms.

In 3D model, the actual physical distribution is not focused on. We care more about relative distribution. Therefore autonomous domain node layer uses a ring layout. From the composition of 3D point of view, if we arrange the router level nodes by means of long-distance and cross-domain distribution, maybe the situation which the nodes and links in large-scale network are in chaotic state occurs. We arrange the node in a relatively domain of its parent node's scope. Meanwhile, in router level, we adopt the random distribution. And the last level, terminal level has the same distribution rules with the router level. Then the next Figure III will show you a clear arrangement.

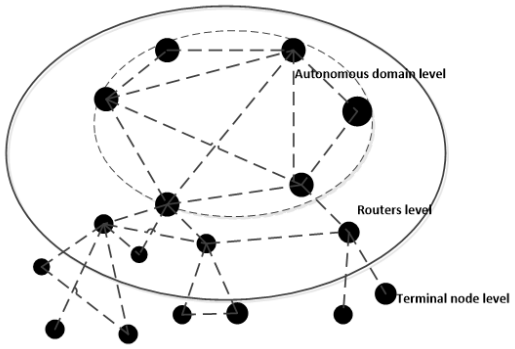

FIGURE III. NETWORK TOPOLOGY

C. *Middleware Node.JS in Communication Model*

Web development mode has a lot of change recently. It experiences Web1.0 era, back-end based MVC era, SPA ear using Ajax, front-end based MVC era. In different eras, the focus of web development is changing. But there is a same goal which reduces the complexity of development and improves code reusability and performance. The emergence of Node.JS redefines separation about front end and back end. Figure IV will show you contact and allocation about front end and back end.

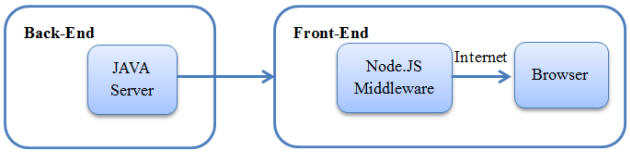

FIGURE IV. BACK-END AND FRONT-END STRUCTURE

As front-end the browser deals with revealing functions. And JavaScript can aim the purpose to interact with users. Based on modular mechanism, Node.JS can provide a lot of functionality module in accordance with specific requirement, for example, data acquisition, application template. As back-end the server deals with logic operation and data updating.

In this visualization system, it not only can show static network topology file, but also can update data visualization effect and static files in real time，which is different from other visualization tools.

But if we read static file and render objects simultaneously in front end, the performance will become a serious problem. The performance problem will probably cause white screen or no response. Considering these problems, we use Node.JS to be middleware, which can provide real-time data transmission and data updating in local files. The detailed communication process is shown in Figure V.

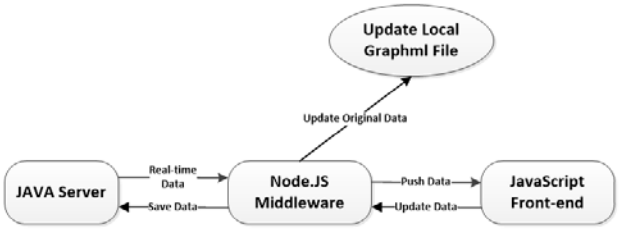

FIGURE V. COMMUNICATION PROCESS

D. *Forward and Backward Communication*

As a library, Socket.IO is an important module which takes charge of internet communication between Node.JS and Web front end. We use polling to be communication mode. Front end use event driving to make connection with Socket.IO. After that, front end can communicate with Node.JS.

1) Forward Communication: JAVA Server processes data and push real-time data to Node.JS. Data format is JSON. Node.JS passes JSON data to front end. The front end uses JavaScript to parse JSON data and update corresponding nodes and links.

2) Backward Communication: Visualization System gets all information which is submitted by users. The front end pushes various data to Node.JS. After that, Node.JS saves the various data into local files. Meanwhile, Node.JS pushes data to JAVA Server by socket to perform data backup.

The detailed communication is described by sequence diagram in Figure VI.

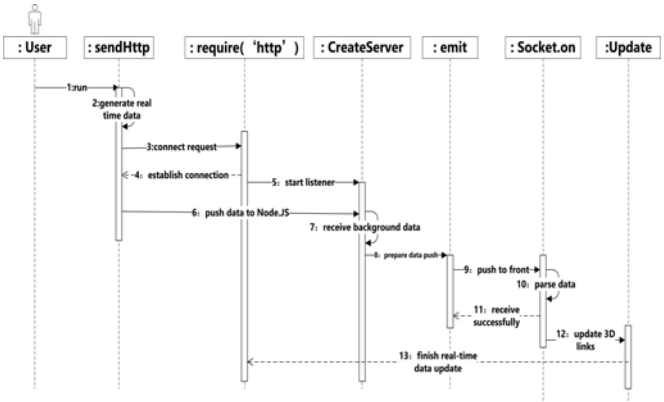

FIGURE VI. DATA TRANSMISSION

E. *Update Local File by Node.JS*

Node.JS file system module provides I/O operations. You can control the module to perform functions to open and write files by using method "require".

XML DOM is a XML document object model, which defines standard methods to process and deal with XML document. GraphML file is based on XML file. So we choose XML DOM to update static and local files.

Function createElement can create new element node, function createAttribute and setAttributeNode can create node's attribute together. Function appendChild adds nodes into file. After that, new nodes will be rewritten back into local text file to update.

V. TESTING

Hierarchical network topology visualization is shown in Figure VII. In this system, users can rotate, zoom, drag and move the whole model to view and understand detailed network information.

Meanwhile, this system also provides interactivity. Users can modify nodes and links' attributes. Users can use highlight and hiding methods to filter out objects which don't comply with specific attribute.

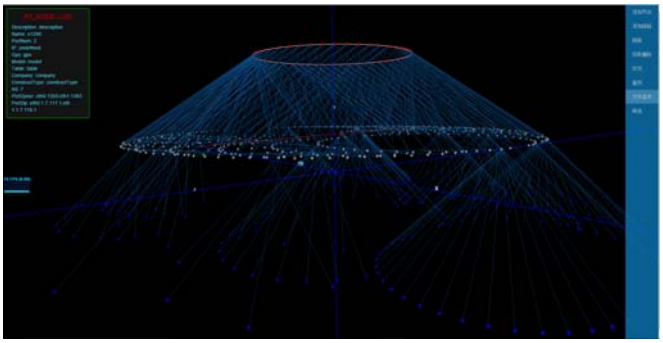

FIGURE VII. VISUALIZATION

Because of cross-platform, the performance of system is almost alike under different platform. So, in this part, we test the system's reliability and response rate under Windows 7.

We select different magnitude of network scale to test the response speed in actual operation environment. The major hardware devices are CPU, RAM and Graphic Card. The CPU is Inter(R) Core(TM) i7-3770 3.40GHz. The RAM is 8.00GB. The Graphic card is NVIDIA GeForce GT 630 at least.

We choose two kinds of network scale. One is 300 nodes and 300 links. The other one is 2000 nodes and 2000 links. Then we test its load time, its fps and performance when it executes scene roaming. The result of comparison is shown in Table I.

TABLE I. COMPARE THE TIME & PERFORMANCE USED OF DIFFERENT NETWORK SCALE RUNNING IN SYSTEM

Network Scale	Load Time	Operation	fps
300 nodes 300 links	27.98s	static	88
		rotate	75
		move	76
		zoom	90
2000 nodes 2000 links	66s	static	18
		rotate	16.5
		move	22
		zoom	15.5

From the table, we can find that the load speed will be slow, even more than 1 min, when we use larger network scale. Mea-nwhile, in the process of running, the fps will become lower with the increase of network scale. But it still maintains smoothly running.

VI. CONCLUSION

In this paper, large-scale network nodes and links, according to the hierarchical layout, are shown to users more clear and beautiful. We not only improve performance and availability, but also realize real-time update and local file modification, which is an extension of network topology visualization tools set. Rendering and displaying massive objects at the same time in terms of efficiency has some development and progress in this paper.

This network topology system confirms that there is on only a solution to cover all 3D data visualization. Cross-platform and cross-browser 3D application based on Web GL will be an effective solution and be widely accepted by users.

ACKNOWLEDGMENT

This work is supported by National Natural Science Found ation of China 61272511, 61370063; National High Technology Research and Development Program of China (863 Program), 2013AA01A212; SRF for ROCS, SEM; Beijing Higher Education Young Elite Teacher project, YEPT1178.

REFERENCES

[1] ZhiCheng Huang. Image data wise information data Visualization Technology and prospects [J].Electronic Vision and Decision , 1999,(6):3-9.

[2] Rohrer R M, Swing E. Web-based Information Visualization[J]. IEEE Computer Graphics and Applications, 1997 (4): 52-59.

[3] Center for Applied Internet Data Analysis Walrus [EB/OL]. http://www.caida.org/tools/visualization/walrus/

[4] National Laboratory of Applied Network Research [EB/OL]. http://www.nlanr.net/

[5] McGregor T, Braun H W, Brown J. The NLANR network analysis infrastructure[J]. Communications Magazine, IEEE, 2000, 38(5): 122-128.

[6] Three.js [EB/OL]. http://threejs.orgs .

[7] Wenwen Tan, Shiyong Ding, Guiying Li. 3D Web Animation Design and Implementation Based on Web GL and HTML 5[J]. Computer Knowledge and Technology, Vol.7,No.28,October 2011.

[8] Tony Parisi. WebGL: Up and Running [M]. O'Reilly Media, Inc, USA,1(2012:40-50).