A PROTOTYPE OF WEB-BASED SIMULATION ENVIRONMENT: USING CGI AND JAVASCRIPT

Tan Kee Leong, Borhanuddin Mohd Ali, Veeraraghavan Prakash, Nor Kamariah Nordin
Department of Computer and Communication System,
Faculty of Engineering, Universiti Putra Malaysia
43400 Serdang, Selangor, Malaysia.
e-mail: keeleon@pc.jaring.my, borhan@ieee.org, prakash@eng.upm.edu.my, nknordin@eng.upm.edu.my

Abstract: In this paper, we present the Websim project, a simulation environment for self-developed simulation tool. It will provide web and distributed features to a simulation model or library created using C/C++ language. Websim is implemented using CGI and Javascript technologies. It can be accessed on any platform (Unix, PC or Mac) using a standard web browser. With this simulation environment, an easier and faster integration of simulation and visualization techniques and tools into the Internet will be supported. The possibilities and problems of the used techniques are discussed in this paper.

Keywords
WWW, Web-based, Simulation, Simulation Environment, CGI, Javascript.

1. INTRODUCTION

The growing global use of the World Wide Web (WWW) and the rising acceptance of Intranet-based systems have a strong influence on simulation, animation, and visualization applications. But currently, most existing commercial and non-commercial simulation tools are typically platform-dependent and are not designed to work in the Internet or an Intranet.

At the Broadband and ATM Research Group [1], Universiti Putra Malaysia, network protocol simulation tools are widely used in several research projects. The popular simulation tools among the researchers here are self-developed C/C++ program, Mil3 Opnet, NIST Network Simulator and Mathworks Matlab. These simulation tools are referred as classical simulation tools and are typically platform dependent. Other limitations of current classical simulation tools are expensive cost, maintenance difficulties and limitation in reusability. This Websim project is concerned with developing a prototype of web environment which provide easy creation of web-based simulation tool to assist the researchers in our research group. Web-based simulation has many benefits in comparison to classical systems, such as platform independence, maintenance minimization, reusability, and interoperability [2]. Meanwhile, web browsers make the Internet a more user-friendly environment by integrating all those resources into a single tool that eliminates the necessity of novice users to struggle with a steep learning curve [3].

2. RELATED WORKS

Most existing Web-based simulation tools have focused mainly on the development of the runtime simulation libraries and mechanisms on the Web, such as Simjava[4], Javasim[5], Silk[6] and the development of the distributed simulation environments such as JavaRMI, CORBA. Some of these tools require simulation model developers to be good at Java and simulation languages. Besides that, web applications developed in Java are relatively slow in speed performance. To ease the burden of modelers, GUI based modeling environments are essential. OPNET[7] and Digital Workshop[8] support the visual development environments, but OPNET could not support Web-based simulation capability, and Digital Workshop could be used only for digital circuit design.

3. DESIGN AND ARCHITECTURE

Websim prototype shows how one could extend the capability of simulation tools into the web. One important property of Websim is the execution of simulation runs on a remote server. It is designed in order to provide distributed and platform independent features to multiple users. For instance, a user could perform the simulation works at home using the simulator located at his office, while another colleague at the office watches the results shown on the computer screen via a web browser.

Websim is developed based on client-server architecture - a central web server coordinates the accessed to local computational resources. The central web server receives client requests and return the results back to the client via HTTP connection. The interaction between the server and client components are shown in Figure 1.

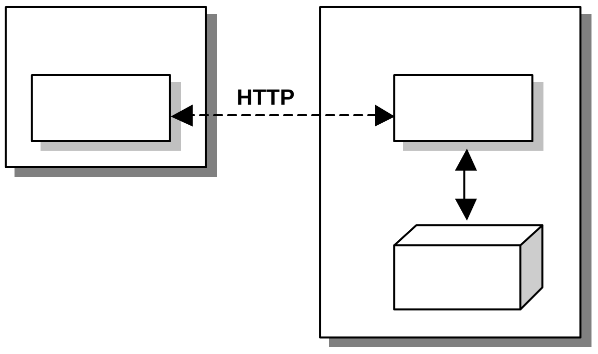


Fig. 1: Client-server interaction in Websim

Websim Architecture

Websim architecture consisted of several modules developed using server-side CGI scripts and Javascript. The CGI scripts, written in Perl process the requests coming from various Websim modules. For example, script related to the Image viewer module opens the simulation project folder, arrange and display all the GIF files with their names in a table format. Meanwhile, the client-side Javascript is used to validate all user inputs before submission to the server site, thus eliminates some of the unnecessary work on the server and tightens the security. For example, we filter the filenames to prevent meta-character like ; or & and check to see if the new directory name conflicts with an already existent directory's name. Most of the time, the server will not send an error message, because of the precautions taken at the client side. The server site ext ensibility mechanism such as CGI and client-side support with Javascript applications open the way to world-wide distributed collaborative learning/teaching environment such as Websim.

Each module is written as one Perl script which perform a specific task. There are three basic modules for both administrator and normal user. But Administrator has two additional modules, which are the 'Upload' and 'Site Manager' modules. We provide also a main menu to integrate all the modules. Below are brief modules explanation.

Upload Module – ***Administrator only***
Administrator may choose this option for step by step instructions on how to create a web-based simulation on the fly. After naming the project, the administrator may upload the simulation (.exe) file, generate the user interface, and customize the settings of output graph.

Site Manager Module – ***Administrator only***
Site Manager is a CGI script that performs most of the file operations required for web site maintenance. It supports edit, delete, rename and upload files.

Image Viewer Module
This module display all GIF images of simulation graphs in a choosen project directory. The images are displayed in a table format and the user may download them to local hard disk.

Image Viewer Module
This module provide a page for the user or administrator to customize the default graph settings. Several properties such as the graph size, lines type, show grid, etc. may be changed and set here.

Run Simulation Module
This module allow the user to run the simulation file, uploaded by the administrator earlier. Once the simulation is done, an the result would be output into a GIF file. The user may choose to keep the file at server or download it.

Websim Actors

There are two types of Websim user: the administrator and normal user. The administrator has the access to create and manage a simulation project. They are granted with upload, change, delete and other administrative services. Meanwhile, normal user is only allowed to view and execute the simulation system. Fig. 2 shows a use case diagram to visualize the relationships between two users and their services.
As shown below, the administrator has two additional services compared to normal user.

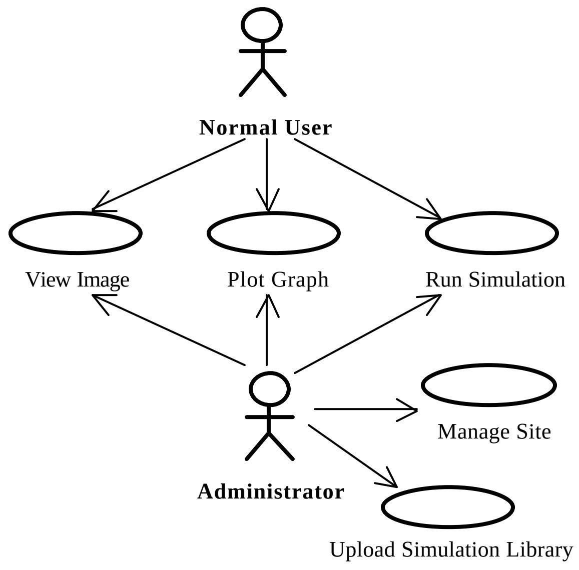


Fig. 2: Relationship between Websim users and their services

4. SYSTEM IMPLEMENTATION

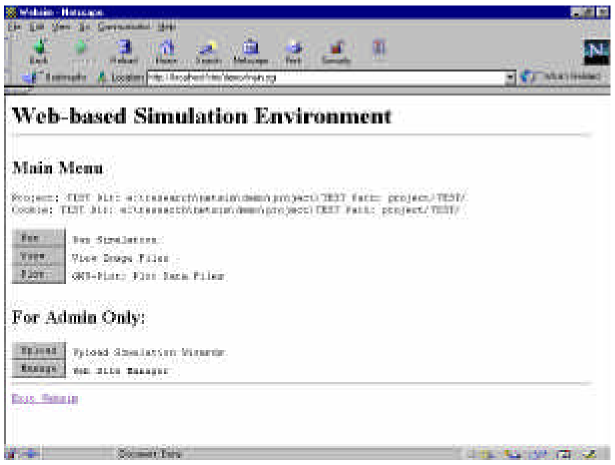
In this section we shall discuss how Websim works by using one simple case scenario. Say that a lecturer would like to create a web based simulation project for Poisson probability distribution function. In this case, the simulation library is the Poisson probability distribution function. The first thing the lecturer has to do is to write a C/C++ program to simulate the function. The simulation library program must follow a certain format required by Websim, and be compiled into a binary format (.exe). Websim does provide a template file for Websim users to refer. Once this is done, next is to deploy the simulation library to Websim server.

Administrator Access

In order to upload the simulation library onto the Websim server, the lecturer must have administrator access. Shown here are the steps on how administrator may upload a simulation library by using the upload service provided by Websim:

- Select simulation library file (.exe).
- Specify simulation parameters.
- Send file to server.
- Specify web interface settings.
- Preview
- Finish

After login as administrator, he may choose either to create a new simulation project or open an existing one. Once the lecturer create/open a project, he will proceed to the Websim main menu, providing all the services for administrator, as shown in Fig.3. If he choose the upload simulation library service, Websim asks for an executable (.exe) file from the administrator and require administrator to specify parameters such as the numbers of input and output parameters for the simulation library. Once uploaded, administrator has to specify the web interface settings for that particular project, such as project title, names of simulation inputs, size of graph to be plotted, etc. Once this is done, this module will preview the web interface and administrator may choose to alter the settings or accept it from there.



Web-based Simulation Environment

Main Menu

Project: TEST Dir: e:\research\[illegible]\demo\project\TEST Path: project/TEST/
Cookie: TEST Dir: e:\research\[illegible]\demo\project\TEST Path: project/TEST/

Run — Run Simulation
View — View Image Files
Plot — GNU-Plot: Plot Data Files

For Admin Only:

Upload — Upload Simulation Wizard
Manage — Web Site Manager

Exit Websim

Fig. 3: Snapshot of Websim main menu

Normal User Access

In this scenario, the Websim normal users are the lecturer's students. After login, users may select a pre-installed simulation project. Once, entered the simulation project page, user may input several parameters to be passed to the simulation library. After simulation is done, the results will be graphed. An independent plotting program, Gnuplot [9] translates the ASCII input text file generated by the simulation system into a GIF image file. The CGI scripts then return an HTML page including the generated image. The implementation components and its flow are shown in Fig. 4.

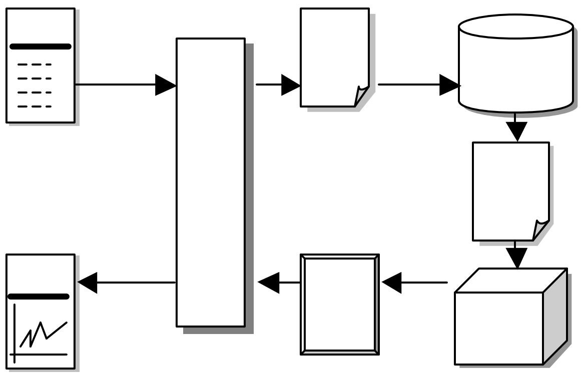


Fig. 4: System Implementation for Normal User

Multi-Users Supports

Websim allow many users to access and perform simulation simultaneously. Hence, in order to have a fair and optimum supports to all users, certain steps have been implemented. We have used variables based on the process ID – ($$ in perl) to create temporary files for each user. The $$ variable refers to the number of the process running this program, as it does in a shell script. The number obtained from $$ is used to create a unique filename, even if multiple instances of the programs run. The process identification is prefixed to each filename.

5. EXPERIMENTAL RESULT AND EVALUATION

We have successfully installed, run and test Websim on two different servers as shown on the table below:

Table 1: Websim experimental testing platform

	Server 1	Server 2
Processor	Pentium II 333	Pentium III 500
RAM	64 MB	256 MB
OS	Windows NT	Windows NT
Web Server	Omnicron Omnihttpd	Microsoft Internet Information Server

A small and simple simulation model, the Poisson Probability Distribution has been used. We were able to upload the simulation file (98 K Bytes) easily and were able to generate the final graph in approximately 6 seconds. Experiments were done within the same LAN as well as via the internet, and under light to medium traffic. In the future, the actual simulation model would be replaced with several self-developed Wireless ATM protocol using C/C++ Language. These models are the results of some Master and PhD. thesis in the Broadband and ATM Research Group. At the moment of this report writing, the integration of these models with Websim components is still under development.

Simulation Library

The simulation files perhaps is the major limitation of this project. Simulation library refers to the simulation executable file created using C/C++ compiler. C/C++ is preferred because of its speed performance. They run 8 times faster than Java, in term of simulation times [10]. But the disadvantage is when writing the simulation program in C/C++, we have to specify a fixed number of input arguments and predefined number of output files. This gives rise to several limitations, such as we could not add or modify the number of arguments, and we have to supply the value of all the input arguments. Hence everything is static. The results in the text file must follow the standard Gnuplot program [9] data files format, such as specifying the first column as *x* and the second column as *y*.

6. SECURITY ISSUES

A general limitation of Websim is simulation run must be operable by a command line executing a batch operation file. From the security perspective, this would expose the web server to some harmful program. Besides that, the input forms of interactions are limited in the usage of CGI techniques in running the simulation model on a remote server.

We have anticipated potential attacks and implement strong security measures. The most important security measure, which has been implemented is to password protect the directory of the simulation project. Accessing the script will require a password from both Websim user and Administrator so that only authorized users will be able to load the form. Besides that, Websim will only browse its own directories below it. This restriction prevents Websim from being able to access any files outside the site. It will also not browse hidden files or directories. Websim also checks for shell escape characters before performing any operations. Finally, Websim will accept form data only from itself. This helps to prevent arbitrary form data from being submitted.

7. CONCLUSION

In this paper we have summarized our experiences in building a web-based simulation environment for the Poisson distribution function. We also described the various components used to develop Websim. The project is still in the last part of development stage and our final goal is to create a general web-based simulation environment which contains a comprehensive collection of simulation library related to our research group. Websim is a unique tool that allows the development and execution of certain simulation model to be performed on the web. It is one of the first prototypes of simulation tools that uses the Web as the standard interface for accessing computational resources. The prototype version of Websim and documentation can be accessed at the following URL: http://cc.eng.upm.edu.my/sim/

8. REFERENCES

[1] Broadband and ATM Research Group, Faculty of Engineering, Universiti Putra Malaysia. http://cc.eng.upm.edu.my

[2] Frank Seibt, Marco Schumann, Jürgen Beikirch, “Concept and Components for a Web-based Simulation Environment (WBSE)”. http://www.isima.fr/scs/wbms/d6/seibtf.html

[3] Kivanc Dincer and Geoffrey C. Fox, "Using Java and JavaScript in the Virtual Programming Laboratory: A Web-Based Parallel Programming Environment" NPAC Technical Report, SCCS. http://www.npac.syr.edu/pub/by_index/sccs/papers/html/0750/abs-0788.html

[4] Howell, F.W., The Simjava Homepage, April. 1999. http://www.dcs.ed.ac.uk/home/hase/simjava,

[5] University of Newcastle upon Tynem Javasim Homepage. http://javasim.ncl.ac.uk/, April, 1999.

[6] Healy, K.J. and R.A. Kilgore, “Introduction to Silk and Java-based Simulation” Proc. Of the 1998 Winter Simulation Conference. 1998.

[7] Mil 3, “OPNET Tutorial Manual”, Washington DC, 1997.

[8] Fishwick, P., Digital Circuit Simulation on the Web. October, 1999. http://www0.cise.ufl.edu/~fis hwick/dig/dlesp.htm,

[9] Williams, T. and Kelly, C., "GNUPlot: An Interactive Plotting Program Manual, version 3.6a". http://www.cs.darthmouth.edu/gnuplot/gnuplot.htm

[10] R.McNab, F.W.Howel, “Using Java for Discrete Event Simulation”, Department of Computer Science, University of Edinburgh.