Proceedings of the 1998 Winter Simulation Conference
D.J. Medeiros, E.F. Watson, J.S. Carson and M.S. Manivannan, eds.

COMMERCIAL SIMULATION OVER THE WEB

Larry Whitman
Brian Huff
Senthil Palaniswamy

Automation & Robotics Research Institute
The University of Texas at Arlington
7300 Jack Newell Boulevard South
Fort Worth, Texas 76118, U.S.A.

ABSTRACT

Modeling and simulation provide objective analysis tools for many fields including manufacturing. This paper presents the requirements and describes the usefulness of a web-based interface to discrete-event simulation. A description of related efforts is first presented and an approach is then described. The approach develops a web-based interface to use commercial discrete-event commercial tools.

1 INTRODUCTION

Modeling and simulation are objective analysis tools. The tools themselves have a high entry price and developing the in-house talent to effectively utilize the tool is even more costly. Many small to medium sized enterprises desire to use powerful analysis tools, but are unable to afford the start up cost. The world wide web is a useful mechanism for providing access to numerous tools. The world wide web is a truly "distributed" environment. This distributed environment facilitates providing tools to small companies. A review of the literature shows three types of web-based discrete-event simulation tools (adapted, revised, and expanded to four types from (Lorenz, Dorwarth et al. 1997)):

Server Hosted Simulation: Existing simulation tools can be hosted on a web server and accessed by clients via normal hyper text mark up (HTML) pages. This type of simulation provides the advantage of using a familiar tool and enables the reuse of existing models. The primary disadvantage is that the client can only provide inputs and view pre-specified outputs at a specified time. However, viewing the outputs at specified times is useful for providing “snap-shot” views to the model. The communication power provided by the animation in these simulation tools is not visible over the web. An example of this is shown in Figure 1.

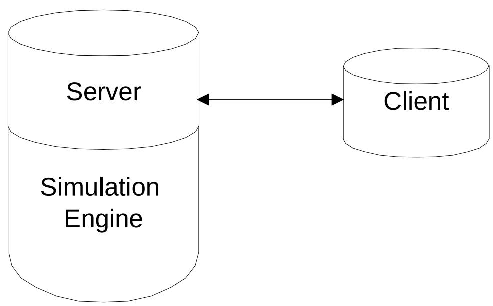

Figure 1: Server Hosted Simulation

Client Executed Simulation: Java, an object-based language, is used to develop simulation code that may be executed on the client machine. Several Java based simulation languages have been developed. These Java simulation packages are fairly new, yet have a rich set of simulation features. The disadvantage is the lack of maturity of the language and few existing models.

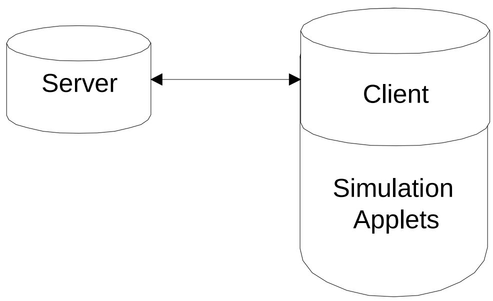

Figure 2: Client Based Simulation

These applications typically require the user to learn a new language. Some simulation tools use an applets based approach that minimizes the learning curve, but also minimizes the power and flexibility of the tool. The performance of this type of simulation is also limited by the client machine performance capability. An example of this is shown in Figure 2.

Hybrid Client/Server Simulation: This type of simulation attempts to combine the advantages of server hosted and client executed simulations. This attempt at the best of breed is achieved by hosting the simulation engine on the server and using Java for visualization of the animation to provide a dynamic view on the client machine. The authors are currently unaware of any practical implementations of this type of simulation although literature shows research in this direction. An example of a hybrid simulation is shown in Figure 3.

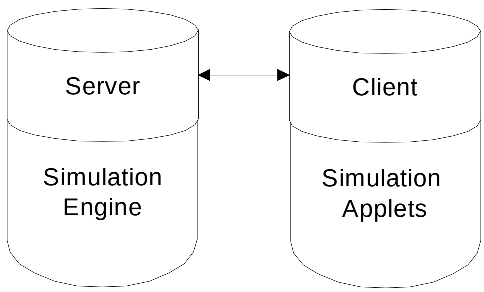

Figure 3: Hybrid Client/Server Simulation

Client-View Simulation Output: This type is a specific instance of server hosted simulation. The client reads a post execution animation trace file. This allows the client to view the animation in a Java enabled browser. The animation capabilities of the state-of-the-art discrete-event simulation tools have become a key benefit for users of simulation. The animation capabilities are frequently the key to selling the concepts identified by the statistical output to decision-makers. Skopeo is an example of an animation tool that provides these animation capabilities (Lorenz and Ritter 1997).

This paper describes two approaches used to perform web-based simulation and then describes an additional approach for interfacing with existing commercial off the shelf tools as the simulation engine. The interface is Visual Basic Script in a common HTML file.

2 REVIEW OF CURRENT WEB BASED SIMULATION

In this section we review the two primary approaches to web-based simulation. Both of these approaches take advantage of the programming language of the internet, Java. One approach develops a new simulation language using Java or in an object based language. The other approach ports an existing simulation language, such as GPSS, and creates it in Java.

2.1 New Java Simulation Languages

The literature reveals several new simulation languages created in Java. A brief description of each of these follows. SimJava (Kreutzer, Hopkins et al. 1997) utilizes the advantages from object-orientation of encapsulation, polymorphism, and inheritance. SimJAVA uses a basic discrete event simulation engine and extends this with Java's graphical user interface features. Silk, (Healy and Kilgore 1997) is a process-based language built in Java. Silk also uses object-oriented concepts to facilitate reuse of components and also takes advantage of the multi-threading capabilities of Java. MOOSE (Cubert, R. M., T. Goktekin, P. A. Fishwick, 1997), although not a Java implementation, does use object-oriented concepts and provides a modeler interface to populate methods and classes. MOOSE also explicitly defines a model repository to encourage model reuse.

2.2 Java Versions of Existing Simulation Languages

A Java implementation of GPSS, JavaGPSS, has been developed. GPSS is a familiar language to most simulation professionals. Most initially learn simulation using GPSS syntax. The primary advantage of this implementation is that JavaGPSS is highly compatible with GPSS/H. JavaGPSS allows a user to enter code in GPSS syntax which is then converted to Java source code. The simulation model is then compiled into Java bytecode. Then the simulation model could be executed on the client machine in a Java browser (Klein, Strassburger et al. 1997).

3 SERVER HOSTED SIMULATION USING COMMERCIAL TOOLS

Having presented the various approaches to web based simulation, we now present an environment for interfacing commercial off the shelf discrete event simulation tools with web clients. This approach has the advantage of using tools that are familiar to many simulation practitioners with literally thousands of existing models developed.

We categorize three advantages to this approach: 1) familiarity with the tool and its syntax, 2) the use of existing models, and 3) executing the simulation engine on a more powerful machine than most client machines. This section will describe the enabling technologies for web based simulation.

Simulation is used primarily as an analysis tool. A model is built to provide an answer to a question. Typically, a series of parameters are varied in an attempt to discover the results in a series of scenarios. These “whatif" games are played until the best reasonable solution is found. To facilitate this type of analysis, variables are placed in the simulation to allow easy modification of key parameters. Many simulation packages provide an additional tool to run simulations several times varying the parameters in order to find an "optimal" solution. Our approach makes use of these variables.

Computers are now so pervasive that most employees use word processing and spreadsheet software in their jobs. Most advanced simulation tools provide mechanisms for interfacing simulation models with standard office software suites like Microsoft Office. Microsoft has developed OLE (object linking and embedding) Automation for this purpose. Someone with little or no knowledge of simulation can execute a simple macro in a spreadsheet and receive only the outputs necessary for making decisions. Boeing uses this approach to allow factory foremen to enter a few parameters in a spreadsheet on a shop floor computer and run a macro. This macro executes the simulation model with the users values assigned to the key variables in the simulation. The spreadsheet then returns the required information. The resultant information in the spreadsheet is used for scheduling (Quinn 1997). Obviously, someone knowledgeable in simulation is required to develop the initial model, but once the model is built and validated, the simulation expert is no longer tied up with the daily use of the program. A working knowledge of spreadsheets is all that is required to use this system.

The development of both client and server side scripting enables the use of these "spreadsheet type" macros over the internet. Both VisualBasic (VBScript) and JavaScript (JavaScript and Jscript) may be used to provide an interface into existing simulation models. The advantage of using these tools is that the spreadsheet macros are written in visual basic. Visual basic has minimal differences with the visual basic scripting edition. Therefore existing spreadsheet macros interfacing with simulation models can now be run over the internet. By adding the proper HTML header and footer information, most visual basic macros in spreadsheets can be ported directly for use on the internet. A summary of the steps required to implement commercial simulation over the web is described in the next section.

4 INFRASTRUCTURE ESTABLISHMENT PROCESS

This section describes the necessary steps to establish the required infrastructure for server hosted simulation. To facilitate understanding, examples will be given for both ServiceModel and WITNESS. Two examples are given to demonstrate the generalness of the process. It is likely that *any* commercial software application that has an object linking and embedding interface will work with this method.

Step 1: Create the simulation model as normal in the commercial discrete event application.

Step 2: Add the appropriate variables to the simulation.
For example, in ServiceModel, use Build:Macro option to set the variable. This allows the user in the simulation to modify a variable in the Simulation:Model parameters option. So, really all that is happening is that the user is changing these parameters over the internet, instead of on the actual machine that the simulation exists.

Step 3: Register the commercial software as a server.

Select 'Start:Run':
  In ServiceModel
    "C:\svcmod4\sm.exe /Regserver"
    In WITNESS:
    "C:\witness\witness.exe /Regserver"

Step 4: Create the VBScript code:

Examples of the key commands used are:

- Set smobject = CreateObject("promodel") which will establish the ServiceModel session in VBScript. For WITNESS use

  Set witobj = GetObject(, "WITNESS.wcl").

- Inputbox is used to get the users parameters. This input box can be a Visual Basic window or it can be an HTML form. An example of a visual basic entry window is shown in Figure 4.

- Smobject.setmacro "(any macro name)", (any value) is used as the variable in the code after the value is retrieved from the inputbox command.

- Smobject.simulate is used to start the simulation for ServiceModel.

- Set witobj = GetObject("WITNESS.wcl") is used to execute the simulation for WITNESS.

Step 5: Get the results.

This is considerably different for the two simulation applications tested.

For ServiceModel:

Register the software as another server to retrieve the results from the application.

Register by executing “C:\svcmod4\RDBsrv.exe” for ServiceModel. Then, get the required values from the RDB file of the model by using:

```
Set RDBobject = CreateObject(“RDBDataserver”)
RDBobject.openfile(“C:\svcmod4\models\mname.rdb”)
```

Note: for the RDB file to be created, run the mname.mod (or whatever file that is to be accessed over the internet) and select ‘yes’ when service model asks if you want to view the results, which will automatically create the RDBfile.

To get specific results or values from the output of the servicemodel, use –

```
IF RDBobject.selectdata(1,1,1, i ,j,k) Then
        Mydata1=RDBobject.GetValue
End If
```

Where;

- i – the value of the # which represents the position of the table to be selected.
- j – the value of the column of interest.
- k – the value of the row which has our variable.
- Mydata1 is an example variable used here.

Complete code to run an example simulation model in both ServiceModel and WITNESS can be found at: http://modsim.uta.edu/websim/.

5 APPLICATIONS OF WEB-BASED SIMULATION

There are numerous applications for facilitating the use of simulation over the web. This section discusses two of the more near term use applications.

A primary disadvantage of web-based simulation is that the user can not view the source of the simulation or even view the simulation while it is running. However, there are many situations where this is not a disadvantage, but rather an advantage. In cases such as in a supply chain, frequently the different users of the supply chain want to retrieve information about the other suppliers to ensure "optimality" among the different components. A web based simulation would allow a simulation of the overall supply chain with each supplier only able to change parameters relevant to their operation. The supplier could enter the parameters and get the results without a view to the entire simulation model. Obviously, the supplier needs a view into the entire supply chain, but the exact details that the supplier should see can be explicitly determined.

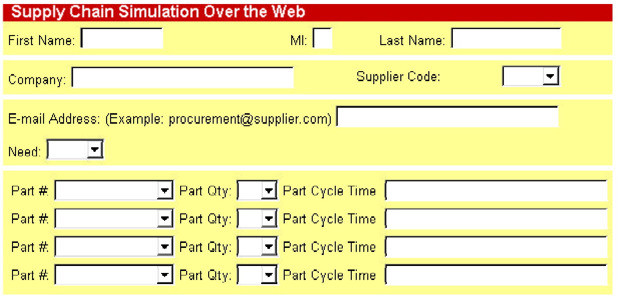

Figure 4: Web Panel from HTML Form

Another application of web-based simulation is for educational purposes. Most students familiar with any kind of coding could take an application and determine the optimal (or a very "good") solution. Again, this type of simulation hides the details, including the application logic from the user. Therefore, an instructor could create a simulation based on principles learned in class and require each student to enter some parameters over the web and record each student's progress in learning the material. For example, a production control class professor could create a factory simulation requiring *kanban* sizes or material requirements planning inputs to schedule a factory. Students could then enter their parameters into the web form and the status of the factory could then be recorded. This is just one of many possible examples for use in a classroom setting.

6 CONCLUSIONS AND FUTURE DIRECTIONS

Simulation is an objective tool providing answers to many manufacturing and other industry questions. The internet enables a distributed environment for utilizing these tools. Many choices are available for executing simulations over the internet. This paper has presented a technique for using commercial discrete event simulation packages over the internet. By utilizing existing interfaces with commercial simulation tools, the transition to web based simulation is simplified. The benefits of web based simulation may be realized by small companies by providing simulation model repositories over the internet. Medium to large manufacturing companies may benefit from the distributed nature over an internal intranet or between supply chain members across the internet.

One commercial package (WITNESS) has recently released the capability to output an AVI file from the simulation which will enable "playback" of the animation portion of the simulation on any windows platform (95 or NT) as well as over the internet. Initial work has already begun examining the feasibility of saving the AVI file and providing a link for the user to “view” the animation after the simulation has been executed. Additional code will be written to present the results in common formats such as graphs and bar charts.

As can be visualized from Figure 4, the authors are currently developing a simulation environment for supply chain design. By providing access to a "black box" simulation to suppliers, each supplier can execute the simulation to determine preset parameters to configure their portion of the supply chain. This paper demonstrates the infrastructure necessary for such an approach.

ACKNOWLEDGMENT

Research for this paper is funded in part by the National Science Foundation sponsored Agile Aerospace Manufacturing Research Center.

REFERENCES

Cubert, R. M., T. Goktekin, and P. A. Fishwick. 1997. MOOSE: Architecture of an object-oriented multimodeling simulation system. In *Proceedings of Enabling Technology for Simulation Science, Part of SPIE AeroSense '97 Conference*, Orlando, Florida, April 22-24.

Healy, K. J. and R. A. Kilgore. 1997. Silk: A java-based process simulation language. In *Proceedings of the 1997 Winter Simulation Conference*, ed. S. Andradottir, K. J. Healy, D. Withers, and B. Nelson, 475-482. Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Klein, U., S. Strassburger, and J. Beikirch. 1998. Distributed Simulation with JavaGPSS based on the High Level Architecture. In *Proceedings of the 1998 international conference on web-based modeling & simulation*, San Diego, CA.

Kreutzer, W., J. Hopkins, and M. van Mierlo. 1997. SimJAVA - A framework for modeling queueing networks in java. In *Proceedings of the 1997 Winter Simulation Conference*, ed. S. Andradottir, K. J. Healy, D. Withers, and B. Nelson, 483-488. Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Lorenz, P., H. Dorwarth, K. Ritter, and T. Schriber. 1997. Toward a Web Based Simulation Environment. In *Proceedings of the 1997 Winter Simulation Conference*, ed. S. Andradottir, K. J. Healy, D. Withers, and B. Nelson, 1338-1344. Institute of Electrical and Electronics Engineers, Piscataway, NJ.

Lorenz, P. and K. Ritter. 1997. Skopeo - A plaform-independant system animation for the W3. In *Proceedings of the Simulation and Automation '97*, Magdeburg, 12-23. Ghent, Belgium: Society for Computer Simulation Europe.

Quinn, T. 1997. Data Driven Simulation. In *Proceedings of the Promodel User's Conference*, Park City, UT.

AUTHOR BIOGRAPHIES

LARRY WHITMAN is a Research Engineer with the Automation & Robotics Research Institute (ARRI) of The University of Texas at Arlington (UTA). He is currently pursuing his Ph.D. degree from the Industrial and Manufacturing Systems Engineering department at UTA. He received his MSIE and BSET degrees from Oklahoma State University. Prior to joining ARRI, he spent ten years in the aerospace industry integrating factory automation and developing and supporting CAD systems. His research interests are in enterprise modeling, simulation, strategic cost justification, and enterprise applications in manufacturing.

BRIAN HUFF is an Assistant Professor in the Industrial and Manufacturing Systems Engineering Department at the University of Texas at Arlington (UTA). He received his Ph.D. and M.S. degrees from UTA. He received his B.S. in Petroleum Engineering from West Virginia University. His research interests are in manufacturing systems design, industrial simulation, industrial automation and robotics, and shop floor production execution and control.