Environmental Modelling & Software 55 (2014) 49–60

Contents lists available at ScienceDirect

Environmental Modelling & Software

journal homepage: www.elsevier.com/locate/envsoft

A client-side web application for interactive environmental simulation modeling

Jeffrey D. Walker*, Steven C. Chapra

Civil and Environmental Engineering Department, Tufts University, 200 College Ave., Medford, MA 02155, USA

ARTICLE INFO

Article history:
Received 19 June 2013
Received in revised form
10 December 2013
Accepted 5 January 2014
Available online 4 February 2014

Keywords:
Water quality
Simulation model
Web application

ABSTRACT

Recent developments in web technologies including evolution of web standards, improvements in browser performance, and the emergence of free and open-source software (FOSS) libraries are driving a general shift from server-side to client-side web applications where a greater share of the computational load is transferred to the browser. Modern client-side approaches allow for improved user interfaces that rival traditional desktop software, as well as the ability to perform simulations and visualizations within the browser. We demonstrate the use of client-side technologies to create an interactive web application for a simulation model of biochemical oxygen demand and dissolved oxygen in rivers called the Web-based Interactive River Model (WIRM). We discuss the benefits, limitations and potential uses of client-side web applications, and provide suggestions for future research using new and upcoming web technologies such as offline access and local data storage to create more advanced client-side web applications for environmental simulation modeling.

Software availability

Product Title: Web-based Interactive River Model (WIRM)
Developer: Jeffrey D. Walker
Contact Address: Dept. of Civil and Environmental Engineering, Tufts University, 200 College Ave, Medford, MA 02155
Contact E-mail: jeffrey.walker@tufts.edu
Available Since: 2013
Programming Language: JavaScript, Python
Availability: http://wirm.walkerjeff.com/
Cost: Free

1. Introduction

Effective management of environmental systems requires understanding the impacts of humans on complex physical, chemical, and biological processes. Engineers and scientists often use simulation models to perform numerical experiments that compare the effectiveness of alternative policies and management strategies for restoring and protecting natural ecosystems. While models have long been an integral part of the environmental decision making process, the abilities of resource managers, policy makers, and interested stakeholders to understand how these models work, what they represent, and their underlying assumptions are limited due to the technical expertise required to develop and use these models (NRC, 1999).

Without direct access to models, decision makers and stakeholders often rely on modeling experts to develop, run, and interpret simulation models on their behalf (Loucks et al., 1985; Booth et al., 2011). This process can create a bottleneck in the flow of knowledge attained through simulation modeling and hinder stakeholder creativity in developing new alternative scenarios because they are unable to directly translate their ideas into the modeling framework (Loucks et al., 1985). As a result, this dependence limits overall stakeholder participation, which is an essential component of the environmental decision making process (Voinov and Bousquet, 2010). Furthermore, this barrier affects not only the ability of stakeholders to directly use and understand models, but also the ability of modeling experts to effectively communicate model results and interpretations with clients and stakeholders.

In the 1980's, the advent of the personal computer (PC) brought a new era to the development and use of simulation models through widely available and inexpensive computing power accessible through new graphical user interfaces. Loucks et al. (1985) predicted that interactive and easy-to-use interfaces would allow policy makers and stakeholders to better understand and utilize models, which in turn would lead to more informed decision making and wider model acceptance. Chapra and Canale (1987) discussed the application of spreadsheet-based models and animated graphical displays to facilitate the communication of

* Corresponding author. Tel.: +1 978 985 5612.
E-mail address: jeffrey.walker@tufts.edu (J.D. Walker).

1364-8152/$ – see front matter © 2014 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.envsoft.2014.01.023

models to stakeholders. While the PC has indeed contributed to advances in environmental modeling over the past 25 years, the barriers between models and non-technical users persist due to insufficient advances in model interfaces, complexity of model software, input data requirements, and difficulties associated with post-processing and interpreting voluminous output data (NRC, 1999; Buytaert et al., 2012). Furthermore, access to desktop-based modeling software can be limited due to licensing restrictions for multiple users as well as incompatibilities between different platforms, operating systems or even software versions. As a result, current desktop-based modeling software is often not only hard to use, but can also be difficult for decision makers and stakeholders to access directly on their own computer.

Over the past 20 years, the World Wide Web has revolutionized human–computer interactions by providing an engaging and accessible platform for users of all technical abilities. Initially, the web was comprised of relatively simple web pages containing text, hyperlinks, and static images. Over time, the evolution of web standards and development practices gave rise to increasingly sophisticated web applications such as email clients, office suites, photo editors, e-commerce storefronts and social networks. The success and widespread adoption of these applications is attributable in part to the design of intuitive user interfaces that are comprehendible to the average web user, as well as increased utilization of client-side approaches to web application development where more of the application code is transferred from the server to the client. By moving the application into the browser, client-side web applications can provide a more interactive and responsive user experience by directly updating web page content from within the browser, and thus requiring less content be generated by the server. For simulation modeling, this client-side approach allows for not only improved user interfaces to access, visualize and analyze the results of models executed on the server, but also the possibility of performing the model simulations within the browser itself.

The use of web applications for the purpose of simulation modeling is sometimes referred to as web-based simulation (WBS) where the primary model interface is accessed through a web browser (Byrne et al., 2010). The three primary architectures for WBS include the remote architecture where both the model simulations and output visualizations are performed on the server, the local architecture where the simulations and visualizations are performed in the browser, and the hybrid architecture where the simulations are performed on the server and visualizations generated in the browser (Byrne et al., 2010). Traditionally, WBS applications were based on the remote architecture due in part to the computational limitations and lack of development tools for supporting browser-based simulation and visualization. For many years, the local WBS architecture could only be implemented using browser plug-ins such as Java[1] applets and Adobe Flash,[2] which provided the tools and features needed to perform model simulations and visualizations in the browser (Byrne et al., 2010). Today, client-side web applications provide a new approach to the local WBS architecture by facilitating interactive model interfaces built using only web standards.

For environmental applications, web-based simulation has been an area of great interest and active research as investigators explore various ways of harnessing the web for model development, execution, and access (Buytaert et al., 2012; Laniak et al., 2013). Many examples of web applications can be found in the environmental modeling literature for discovering and accessing data (Goodall et al., 2008; Huang et al., 2011), performing web-based simulations (Lim et al., 2001; Bacu et al., 2011; Booth et al., 2011; Feng et al., 2011; Goodall et al., 2013; Sun, 2013), and evaluating model uncertainty (Bastin et al., 2013). The majority of this existing research has primarily relied on server-side approaches where both the application and model are executed on the server. Some recent research has demonstrated the hybrid WBS architecture where a client-side web application is used to provide an interactive user interface for visualizing model results that are generated on the server (e.g. Booth et al., 2011). However, the use of client-side applications in conjunction with the local WBS architecture to perform both the simulation and visualization solely within the browser has yet to be explored.

In this paper, we demonstrate the use of modern web technologies and development practices to create an interactive, client-side web application for a water-quality simulation model of dissolved oxygen in rivers and streams. This work illustrates not only the potential for client-side web applications to provide a modern and interactive user interface, but also the possibility of implementing the local WBS architecture by performing the simulation and visualization entirely within the browser using only web standards. In the next section, we begin with a brief review of web-based simulation and discuss the trade-offs between alternative client-server architectures. We follow with a summary of current web standards and development practices for creating modern client-side web applications. We then demonstrate how these technologies can be leveraged to design and implement a client-side application using the local WBS architecture where the browser performs both the numeric computations and output visualizations, and the server simply acts as a repository for storing model configurations as well as a platform for collaboration. We conclude with a discussion on the benefits, limitations and potential uses of client-side web applications for both local and remote execution of simulation models in general, and suggest ways of incorporating new and emerging client-side web technologies to support more complex simulation models.

2. Overview of web-based simulation

Web-based simulation (WBS) refers to the use of web technologies to develop, execute, and analyze simulation models where the primary interface is accessed through a web browser (Byrne et al., 2010). Not long after the web was introduced in the early 1990s, simulation modelers began discussing the potential advantages and limitations of using the web as a platform for simulation modeling (Fishwick, 1996). Byrne et al. (2010) provide a comprehensive review of WBS including the advantages and disadvantages of various client-server architectures. Unlike traditional desktop simulation software where the entire application is installed and executed in a single environment (i.e. the user's local computer), web applications involve two (or more) physically separate systems: the client and the server. Different architectures for WBS applications are often characterized by how the components of the application are divided between these two systems.

Byrne et al. (2010) define three primary WBS architectures – local, remote and hybrid – based on which system the simulation engine and visualization renderer are each located and executed. A local architecture is primarily client-based whereby both the simulations and visualizations are executed within the user's web browser; the server simply provides the initial application code to the client and stores model configurations, input and output datasets, and other persistent data in a remote database. In contrast, a remote architecture is primarily server-based with the simulation engine and visualization renderer both executed on the server; the client then plays a minor role in displaying the text, data

[1] http://www.java.com, accessed June 19, 2013.

[2] http://www.adobe.com/flash, accessed June 19, 2013.