17th ASCE Engineering Mechanics Conference
June 13–16, 2004, University of Delaware, Newark, DE

EM2004

A STATEGY FOR WEB-BASED MODELING OF HYDRODYNAMIC PROCESSES

Akm Saiful Islam[1] and Michael Piasecki[2] (Member, ASCE)

ABSTRACT

Hydrodynamic model generally deals with enormous amount of data and utilizes huge computational resources for simulation. Powerful and robust servers with extensive storage capabilities are therefore desirable for rapid simulation. Unfortunately, it is not always possible for an individual to effort those kind of facilities whereas a central computer system can be the viable alternative to serve many clients. The simplest way for a client to communicate with the central simulation server can be using a web browser and such kind of simulation has been classified as web based simulation. In this paper, simulation of a hydrodynamic model has been investigated as a case study of the large scale application of web based simulation. Standardized description of the hydrodynamic model data or, metadata has been created using geographical information metadata, ISO 19115:2003 standard. A formal specification of the simulation domain or ontology has been developed to share and retrieve this information unambiguously. Ontology based simulation can also be applied for analyze and future reuse of the simulation domain knowledge. Therefore, web based simulation of hydrodynamic models could be a new paradigm shift in the hydrodynamic modeling as well as numerical modeling area.

INTRODUCTION

Numerical models have been used in many scientific disciplines to better understand the physical behavior of the nature. A number of hydrodynamic models are now available to investigate complex flow phenomena of rivers, estuaries, lakes or coastal regions. Despite recent advances to include 3-D representations of the simulation domain and the ever improving level of graphical display options, the modeling user community still faces some shortcomings when faced with the need to move data around between pre- and post processors and to exchange model data in the user community. Typical problems include the lack of a standardized framework to describe model data, inadequate model data exchange facilities, insufficient interoperability of models among different platforms or operating systems, inefficient search and

[1] Department of Civil, Architectural & Environmental Engineering, Drexel University, Philadelphia, PA 19104, USA,

[2] Corresponding author: Department of Civil, Architectural & Environmental Engineering, Drexel University, Philadelphia, PA 19104, USA email: Michael.Piasecki@drexel.edu fax: 215-895-1363.

retrieval of modeling information, and the absence of the possibility to share and reuse the knowledge already gathered about a certain simulation domain. In addition, several scientific communities have recognized the need to develop numerical models that will serve long term objectives, are not proprietary but based on open source components, and should serve as a resource to the community for scientific exchange and further improvement. This has spawned the idea to develop community models. There are many relatively sophisticated community models available such as the groundwater community model MODFLOW (USGS, 2000), atmospheric community models MM5 (MM5, 2004), and community climate system models CCSM (CCSM, 2004), to name just a few. These models are freely available as a modeling tool within the respective community and allow scientists to focus on their needs rather than building a model from scratch (House, 2000). However, most of these community models are not yet designed to operate in an integrated environment that would ease the work burden that comes with the need to share and exchange modeling data within the community. Consequently, the next step for the development of numerical models should focus on a standardized data description, an improved functionality that permits better sharing of both codes and model data, and provide a platform for preprocessing, execution, and retrieval of simulation results in an environment that is operating system independent.

Web based simulation, WBS, was introduced in mid 90's when the web browsers became available (Miller, Fishwick, Taylor, Benjamin, & Szymanski, 2001). The WBS concept is based on the idea that any user can perform a simulation either in the client machine or on the server machines using a web browser. Moreover, it can potentially utilize a wide range of databases and information systems through the web. Because one of the priorities is to build a platform independent system, Java has been recognized as the essential language for WBS (Kuljis & Paul, 2001). Several WBS environments have been developed based on Java such as JSIM (Miller, Seila, & Xiang, 2000), simjava (F. Howell, 1988), DEVSJAVA (Sarjoughian & Zeigler, 1988) to this date. Unfortunately, most of these environments are either currently unavailable or have not been further developed. Wiedeman (2001) conducted a review of these systems and found that most of them were used for test scenarios only and that actual user requirements were not taken into account. We have found only very few WBS systems that permit data I/O operations of several 100Mbytes of data one of which is the Websim3D system (Olsen, 2003) that permits fast access to view and download earth quake simulation results.

In recent years the development of web based technology has seen much progress. The World Wide Web Consortium, W3C, is currently working on the future vision of the WEB, known as "Semantic Web", to build a machine understandable meaningful form of web resources (Berners-Lee, Hendler, & Lassila, 2001). One of these efforts has led to the creation of the Web Ontology Language (OWL), a language that can formalize the domain knowledge with explicit specification in a machine understandable format. Additionally, the formal specification of the domain knowledge in an ontology permits much better retrieval, share, reuse or analyzing of this knowledge. In the following sections we will outline how we have incorporated these new technologies in the development of a WBS for hydrodynamic processes.

WEB BASED SIMULATION OF HYDRODYNAMIC MODEL

Typically a hydrodynamic model requires the I/O of a substantial amount of data such as water elevation, discharge, dispersion data, wind effect data, roughness, viscosity data, boundary

and initial condition data, all of which may be spatially and temporally invariant. We have classified these data into two categories: (1) geospatial data, and (2) model data. Geospatial data includes maps, the numerical grid, the bathymetry, and the digital elevation model, while model data includes the state variables and all coefficients and constants, which are georeferenced to the geospatial data sets. In addition, parameters such as gravity, iteration counters, tolerance limits, to name just a few, have also been included in the model metadata. A multi layered data model has been developed to handle the model data (Fig.1), for which the geospatial data set serves as the basis. Each layer represents a snap shot in time of the model data for a specific time, i.e. time evolution of a specific variable is stacked in layers above the base layer.

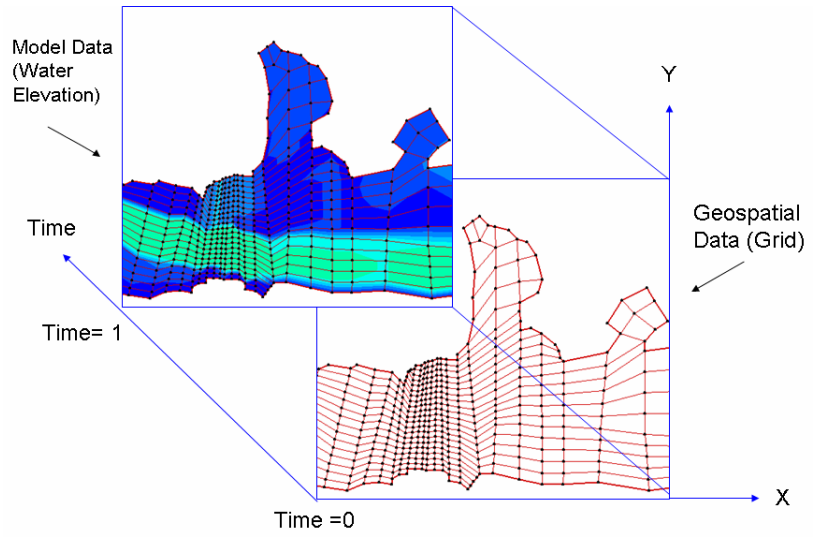


FIG.1. Layer data model of the hydrodynamic model data

This study uses a two dimensional, vertically averaged, finite element code for the numerical integration of the governing shallow water equations. The formulation is second order accurate in time and 5^{th} order accurate with respect to numerical dispersion and diffusion in space. The model was originally developed by Katopodes (1984) and has since been applied and tested in a number of applications; see for example Piasecki and Katopodes (1997). The code has been applied to a test bed that encompasses the Upper Potomac River around Washington D.C. It is comprised of ~1400 nodes and extends over a domain approximately 23 kilometers in length. The flow in this part of the Potomac is primarily driven by the tide signal, which results in significant flow reversals making this domain a highly dynamic and unsteady flow regime.

In order to find an unambiguous description for model data, metadata or “data about data”, must be defined that encompasses and incorporates controlled vocabularies (Bossomaier & Green, 2001). The International Standard Organization, ISO, has published a metadata standard, i.e. the ISO 19115:2003, to describe digital geographic data (ISO, 2003). Since a hydrodynamic code solves the governing equation on a geo-spatial reference, i.e. a numerical grid, the ISO

standard is well suited to be used as a first cut to properly describe numerical model data. However, while the ISO 19915 standard contains many metadata elements, it only represents a generic backbone for use in the geographical data realm. In other words, the standard typically needs to be extended or tailored to the needs of a specific community or application. In addition, it is desirable to create a framework within which not only humans but also machines can deduce meaning between the descriptive elements. This requires the inclusion of a concept that permits machine understandable specifications, like an ontology, to be interpreted by inference engines. One such technology, published by the World Wide Web consortium, is the Ontology Web Language, or OWL, which has been used to encode the ISO 19115 standard (Islam *et al.*, 2004).

WBS-DOMAIN/SIMULATION ONTOLOGY

Using OWL, an ontology was created to describe a numerical model (Fig.2). Geospatial data encompasses the numerical grid, maps, costal boundaries, charts, or a digital elevation model all of which are represented as "*GeoSpatialData*" class in the ontology. The most important geospatial data set is the numerical gird, which is represented as "Grid" class in the ontology. Numerical grids have nodes and elements, which are represented as "Node" and "Element" class in the ontology, respectively. Model data includes data, which is related to geospatial data and is represented as "*ModelData*" class in the ontology. In this instance, model data includes but is not limited to wind, discharge, velocity, water elevation, viscosity coefficient, boundary types, Manning's roughness coefficient, dispersion coefficients, tidal elevation, and tributary discharge. These model data have been represented as a subclass of the "*ModelData*" class in the ontology and could consist of thousands of individual data components. These individual data components are represented as "*Data*" class in the ontology and are connected with either the "*Node*" or "*Element*" class of the ontology.

Besides the necessary I/O organization of data and the description of the flow of these data streams, metadata about the model execution itself needs to be incorporated. These include more general descriptions about the purpose of the simulation, time intervals, modeler in charge, execution times associated with the run, and so on. To this end the WBS ontology has been incorporated into the OWL encoded ISO 19115 to create the foundation of metadata classes. Based on the root class of the ISO metadata ontology ("*MD_Metadata*") three subclasses were created in the WBS ontology: (1) "*MetadataModel*", (2) "*MetadataGeoSpatialData*", and (3) "*MetadataModelData*". In the WBS ontology, each geospatial data type has a metadata class "*MetadataGeoSpatialData*". Similarly, each model data type has a metadata class "*MetadataModelData*". A numerical model could have some model parameters such as gravity, degree of freedom, number of iteration, tolerance limit etc. These parameters are represented as "*MetadataModel*" class in the ontology.

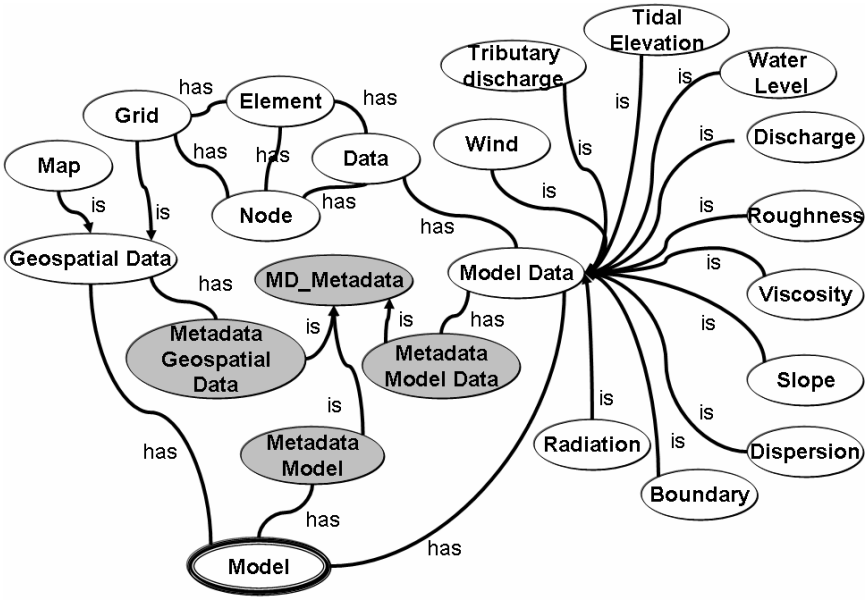


FIG.2. Ontology for a web based hydrodynamic model

WEB BASED SIMULATION ARCHITECTURE

Because of the potentially large amount of necessary data transfers between client and server, we have adopted a client-side-request and server-side-simulation approach as execution mode. This will limit the communication between client and server and therefore omit a potential data transfer bottleneck. The best architectural design for of this kind of system was recommended by Kurniawan, (2002) who suggests to use a Model-View-Controller (MVC) architecture. We will discuss the details of the MVC architecture and how it is adopted for creating a GUI to display different component of the WBS environment in the next subsection.

GRAPHICAL USER INTERFACE (GUI)

The performance of large simulation systems can be improved if the model or business logic is separated from the model views or presentation logics. This design pattern has been known as model view controller (MVC) architecture, which suggests to divide the system into three components (1) model, (2) view and (3) controller (Burbeck, 1992). A model is the representation of the simulation domain; views are the visual representations of the model; and controllers handle the user interactions with the model. MVC has been used in many software development projects and provides the fundamental basis for the JAVA SWING API to support graphical user interfaces (GUI) and graphics functionality (Stelting & Maassen, 2002). We adopted the MVC as the basic architecture of our simulation environment whose components are shown in Fig.3. An object data model, which is based on OWL ontology, is the core of the system. The *Model* stores and retrieves data from an object relational database, PostgreSQL (PostgreSQL, 2003). The *Model* also keeps track of each registered view and will notify its *View* components if any change

in the *Model* occurs. These *View* components are built using Java Server pages (JSP) and contain visual components such as Java Applet or, HTML Form elements. During the session *View* components retain a state of the *Model* to receive data and also contain one or more controllers. Any request for the changes in *View* component will be sent to the controller (which is a Java Servlet program) that contains a reference of the *Model*. Any request fro change from the *View* will prompt the controller to update the *Model*. The controller also decides which view will be displayed to the user. The MVC architecture gives maximum flexibility to the WBS system so that any view can be added or deleted from the system without having to change the model.

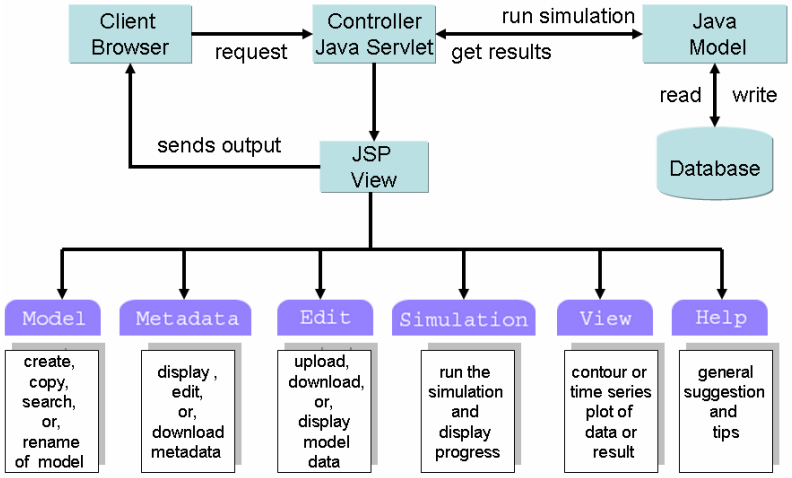


FIG.3. WBS user interface based on Model View Controller (MVC) architecture

Currently six different views are registered to display the different components of the WBS system: (1) model view, (2) metadata view (3), edit view (4), simulation view, (5) graphical view and (6) help view. The Model view is used to search for any pre-existing model. This search can be performed in simple-mode through model description keywords or with a more advanced mode via metadata elements. Once the desired model has been found, the user can copy, rename, or delete it. The Metadata view is used to create, display, edit, or delete metadata for the hydrodynamic model. In Graphic view mode the user can display model results as a contour plot, (Fig.4). The Graphical view also supports the display of temporal model data as a time series plot for a specific point. The Edit view mode permits the display, creation, edit or deletion of model data. The Simulation view tracks the execution of the model for the desired length of the simulation. Finally, the Help view provides suggestions and tips to the user about the WBS environment.

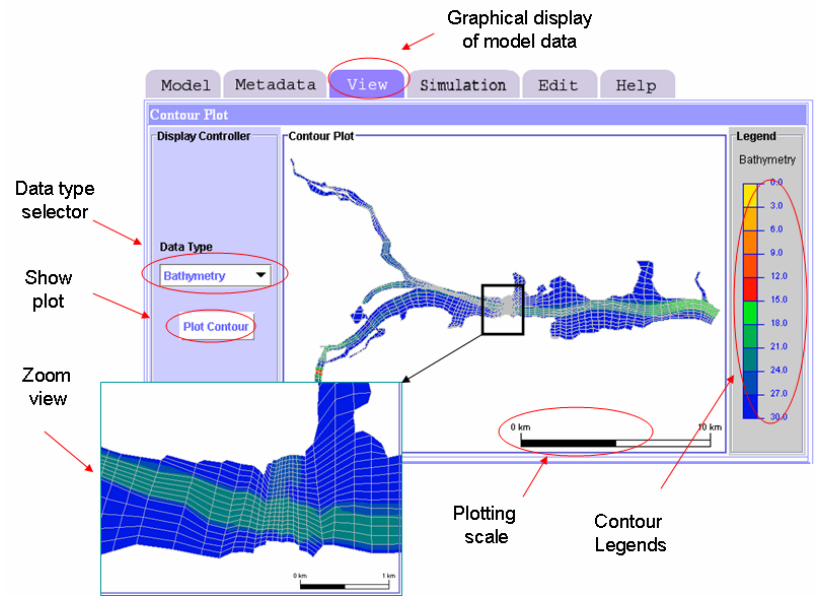
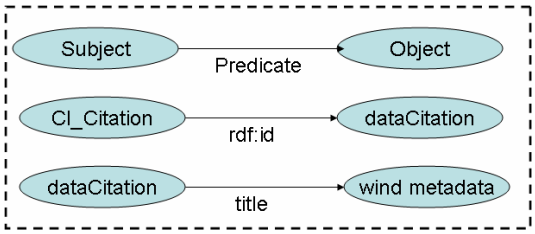


FIG.4. Screen shot of contour plot for bathymetry data using Java Applet

REPOSITORY FOR MODEL DATA AND METADATA

We are using Jena, a Java framework for developing semantic web applications developed by HP Labs semantic web research (Jena, 2003). Jena is an open source package that has an OWL Application Programming Interface (API), and that also supports rule based interface engines. Jena has been used to create instances for model data and model metadata that are based on the simulation ontology. The data can either be stored in a flat file system or in a database. For large data sets, however, storage in databases is preferable because querying and retrieving is more efficient when compared to using a flat file system. For this reason, we chose the object relational database, PostgreSQL, for storage of the ontology instances created by the Jena API. The instances of the simulation ontology are expressed as Resource Description Framework (RDF) in either RDF/XML format or N-3 triple format. Although RDF/XML is the official serialization technique for RDF data, the N-triple format is preferable when using medium to large databases (Jena, 2003). Figure 5 shows an example using a RDF graph model that depicts the serialization and storage of RDF triples in a database. For example, the ontology class "*CI_Citation*" has a property "*title*" to describe a dataset. If "*CI_Citation*" has an instance "*dataCitation*", it can be represented in the RDF graph using the prefix "*rdf:id*" (Fig.5(a)). RDF/XML serialization of this RDF graph is shown in Fig.5(b), while Fig.5(c) shows how this RDF/XML can be stored in a relational database with tables that contain subject, object and property



(a)

```
<CI_Citation rdf:id="dataCitation">
<title>wind Metadata<title>
</CI_Citation>
```

(b)

Subject	Predicate	Object
CI_Citation	rdf:id	dataCitation
dataCitation	title	wind metadata

(c)

FIG.5. (a) RDF triple graph, (b) RDF/XML document, and (c) database table

SUMMARY

This paper introduces the concept for a web-based simulation system for solving the governing equations for a two dimensional hydrodynamic model. The main thrust of this study is to provide a new paradigm for remote clients to explore the computational facility of powerful servers to simulate large scale hydrodynamic processes. To this end we have defined a specific metadata set that describes hydrodynamic model data based on the ISO 19115:2003 metadata standard. In addition we have developed simulation ontologies to search and retrieve this metadata information and to reuse it as simulation domain knowledge. A Jena API has been used to create instances of this simulation ontology and to store these instances in relational database. We have furthermore used the Model View Controller (MVC) approach as core architecture to develop user interfaces which separate the business logic from its representation. The numerical code has been wrapped into a Java environment and is placed on the web Server, which in turn runs the Tomcat server which utilizes technologies such as Java Servlet and Java Server Pages (JSP). The client side graphical user interfaces (GUI) implements Java Applet or HTML forms to display different presentations (views) of the model.

Although this study shows that it is possible to execute large scale numerical model simulations using a web based simulation architecture, there are still many challenges to overcome in case more complex numerical models are targeted for WBS. A possible future direction of the study could be the development of a web based system for three dimensional water quality or subsurface transport models, or the development of efficient search facilities using software agents, and building robust systems using Enterprise Java Bean (EJB) technology for enterprise applications.

REFERENCES

Berners-Lee, T., Hendler, J. A., & Lassila, O. (2001). The Semantic Web. *Scientific American, 284*(5), 34-43.

Bossomaier, T. R. J., & Green, D. (2001). *Online GIS and spatial metadata*. New York: Taylor & Francis.

Burbeck, S. (1992). *Applications Programming in Smalltalk-80(TM): How to use Model-View-Controller (MVC)*. Retrieved 03/05/04, 2004, from http://st-www.cs.uiuc.edu/users/smarch/st-docs/mvc.html

CCSM. (2004). *Community Climate System Model*, from http://www.ccsm.ucar.edu/

F. Howell, R. M. (1988). *simjava: a discrete event simulation library for Java.* Paper presented at the 1988 International Conference on Web-Based Modeling & Simulation, The Society for Computer Simulation International, San Diego, CA.

House, C. (2000). *Report of the Community Sediment Transport Modeling Workshop*, from http://walrus.wr.usgs.gov/transport/

Islam, A. S., Bermudez, L., Fellah, S., Piasecki, M. (2004). *Share and Reuse of Numerical Model Data for the Semantic Web*, from http://loki.cae.drexel.edu/~wbs/ontology/

ISO. (2003). *Geographic Information - Metadata*.

Jena. (2003). *Jena Documentation*, from http://jena.sourceforge.net/documentation.html

Katopodes, N. D. (1984). A Dissipative Galerkin Scheme for Open-Channel Flow. *Journal of Hydraulics, ASCE, 110*(4), 450-466.

Kuljis, J., & Paul, R. J. (2001). An appraisal of web-based simulation: whither we wander? *Simulation Practice and Theory, 9*(1-2), 37-54.

Kurniawan, B. (2002). *Java for the Web with Servlets, JSP, and EJB.* Indianapolis, USA: New Riders Publishing.

Miller, J. A., Fishwick, P. A., Taylor, S. J. E., Benjamin, P., & Szymanski, B. (2001). Research and coomercial opportunities in Web-Based Simulation. *Simulation Practice and Theory, 9*(1-2), 55-72.

Miller, J. A., Seila, A. F., & Xiang, X. (2000). The JSIM web-based simulation environment. *Future Generation Computer Systems, 17*(2), 119-133.

MM5. (2004). *Community Model*, from http://www.mmm.ucar.edu/mm5/mm5-home.html

Olsen, K. B. (2003). *Websim3d: A Web-based System for Generation, Storage and Dissemination of Earthquake Ground Motion Simulations.* Paper presented at the 2003 AGU Fall Meeting, San Francisco, CA.

Piasecki, M., & Katopodes, N. D. (1997). Control of Contaminant Releases in Rivers and Estuaries Part I: Adjoint Sensitivity Analysis. *Journal of Hydraulic Engineering, ASCE, 123*(6), 486-492.

PostgreSQL. (2003). *PostgreSQL 7.4.2 Documentation*, from http://www.postgresql.org/docs/

Sarjoughian, H. S., & Zeigler, B. P. (1988). *DEVSJAVA: Basis for a DEVS-based collaborative M&S environments.* Paper presented at the 1988 International Conference on Web-Based Modeling & Simulation, The Society for Computer Simulation Intnl., San Diego, CA.

Stelting, S., & Maassen, O. (2002). Applied Java Patterns. California, USA: Sun MicroSystems Press.

USGS. (2000). *MODular three-dimensional finite-difference ground-water FLOW model--original model (MODFLOW)*, from http://water.usgs.gov/nrp/gwsoftware/modflow.html

Wiedemann, T. (2001). *Simulation application service providing (SIM-ASP).* Paper presented at the Winter 2001 Simulation Conference, Arlington, VA, USA.