Web based Collaborative Visualization of Distributed and Parallel Simulation

C. Bajaj and S. Cutchin
Department of Computer Sciences and TICAM
University of Texas, Austin, TX 78712
http://www.ticam.utexas.edu/CCV

Abstract

This paper presents an interaction model to support collaborative scientific visualization. Relevant prior work is presented to contextualize the model and its import. An implementation of the model is presented within a collaborative system that supports flexible collaborative coupling of multi-user applications. An example application is presented to demonstrate the capabilities of the model. The implementation is Web based, fully supports multi-user interfaces, uses VRML and compresed VRML for three dimensional graphic display, and is implemented in Java with CORBA support for external server access. An example experiment involving multiple users is described.

1 Introduction

Scientific visualization is the process of using computer graphics to view or explore 3D objects, natural phenomenon, or complex data[18]. It has a history of significant contribution to many areas in science and engineering. Collaborative scientific visualization enhances this process by allowing remote users to interact in the visual study of these phenomena. This enables remote users to contribute more quickly and easily their understanding and views of the phenomena to all members involved in the investigation. Another significant advantage is that collaborative visualization allows for the pooling of remote computational resources, allowing more users to take advantage of a larger number of machines to aid in their investigations. The current widespread deployment of network hardware and software technologies has created an opportunity and need for new collaborative methods and techniques in the creation of middle-ware infrastructure to support sophisticated collaboration over wide area networks.

We envision an environment where multiple distributed users routinely use computational resources located at many different geographic locations. These resources can be remote computation engines, scientific instruments, persistent data sets, and complex analysis systems. The resources are located anywhere and are all accessible via the Internet. Computation times are long, data sets are extremely large and analysis engines routinely access remote data. Such an environment makes it dificult for users to work interactively together both because the problems do not lend themselves to interactive operation but also because distributed users can have vastly different personal schedules and could even be located in different timezones. The problem then is how to provide users with an easy and intuitive mental map of the various resources, users, and tasks that the environment provides and supports. This map must make it clear to users what resources are available, what resources are commited to which users and tasks, and the current state of these resources. The map must must also provide intuitive ways of specifying collaborative tasks, shared activities and still maintain a clear understanding of system activity in users minds. The model must support the users' widely varying schedules and the extensive running time of computations within the environment. We refer to the users' mental knowledge of the state of the environment as collaboration awareness. In this environment we wish to provide users with resource, user and task awareness.

This paper presents an interaction model that provides users with this intuitive map of the resources, users and tasks within the distributed environment. The model defines loops of computation called DSAV loops that support interactive user control of distributed simulations. This loop model provides an easy to understand organization of distributed and cooperative tasks, as well as creating collaboration awareness in users' minds. The loops provide a natural naming scheme for resources, tasks and projects, as well a natural scheme for the display of collaborative tasks, users, and activities. The naming and display schemes provide users with the needed collaboration awareness to effectively engage in collaborative scientific visualization within a highly distributed environment. This model has been implemented as a component of a Web and Internet based substrate designed for collaboration over wide area networks (WAN). An example application implemented using this substrate is presented to demonstrate the capabilities and effectiveness of the model. The example application is constructed using our Internet based collaboration toolkit called Shastra. Several new techniques for flexible application coupling, object sharing, and efficient data location have been developed specifically to support synchronous and asynchronous collaboration on WAN.

The rest of this paper is organized as follows: An overview of important prior work is presented in Section 2. An explanation of the new interaction model is given in Section 3. The important new collaboration features provided by the Shastra substrate are presented in Section 4. Also, a reusable web based visualizer with WAN support is pre-

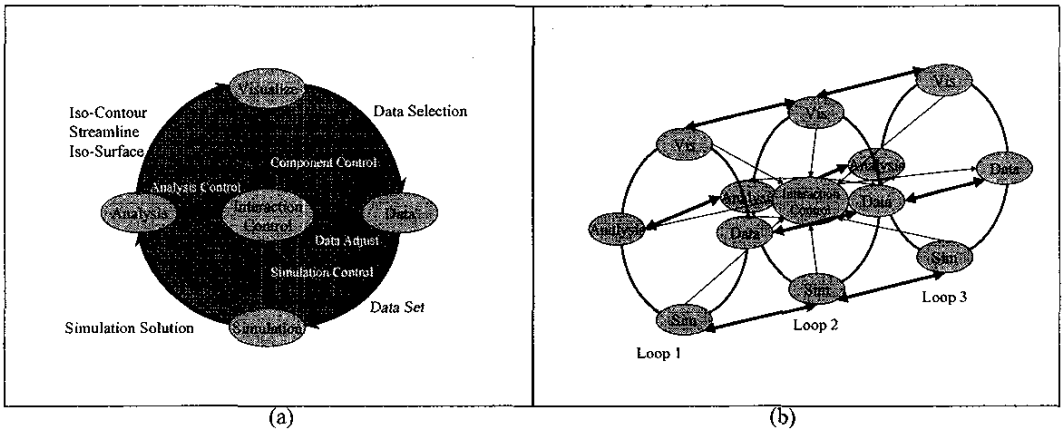


Figure 1: A data, simulation, analysis, visualization (DSAV) loop which displays the communication flow between components is shown in (a). (b) displays the collaborative version of the DSAV loop, showing shared loops involving inter-loop and intra-loop connections.

sented in that section. Section 5 and Section 6 present a soil contaminant simulation and an example user scenario.

2 Prior Work

The collaborative visualization work in this paper draws on past research into interactive computational steering, web based collaboration systems, collaborative synthetic environments, and collaborative visualization. Within scientific visualization three areas need to be addressed to implement efficient visualization: computation, display, and querying. A system that supports collaborative scientific visualization must provide collaboration awareness and ensure that the techniques used do not reduce the efficiency of the methods used to resolve the three problems within a standard system. This is the challenge of collaborative scientific visualization. A framework for collaborative modelling and simulation has been proposed by Sierhuis and Selvin[20]. Several non-general hand-built web capable collaborative visualization systems have been developed such as CEV[8], and DOVE[1]. In Wood[22] several potential scenarios for collaborative visualization are identified and explored. Similar work in computational steering for visualization identifies requirements for WAN based interactive collaborative visualization[17, 13].

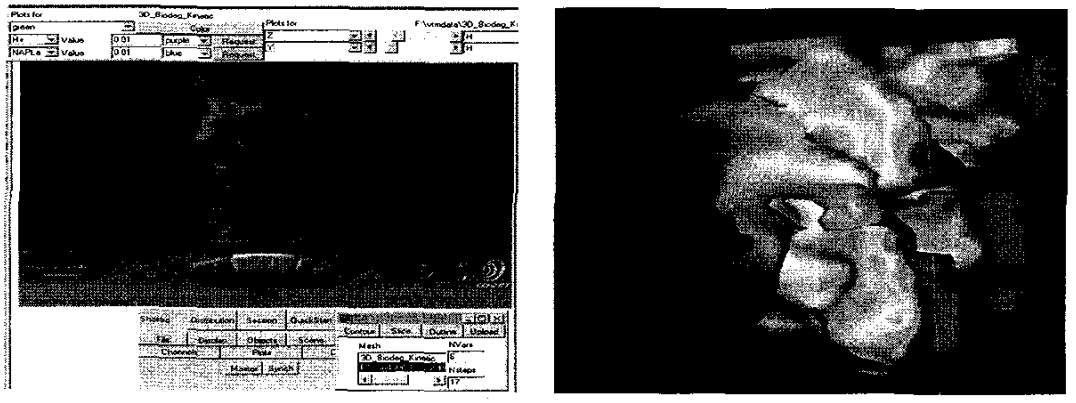
Dynamic program steering is defined in Vetter[21] as two tasks: monitoring program or system state(monitoring) and then enacting program changes made in response to observed state changes(steering). Interactive program steering occurs when humans directly control the steering of the program state. Instrumentation of applications is necessary to allow external programs and users to control the behavior of the computation. Many methods exist for interactive program steering, from simple scripting languages to powerful graphical interfaces.

Flexibility and performance are identified as key requirements for collaborative scientific visualization in [1]. Several projects are underway to extend the capabilities of the Web to support collaborative activity[7, 12, 16]. Collaborative Internet systems are becoming available such as Habanero[9], Tango[6], Infosphere[10], Knitting Factor[5], and GroupKit[19]. These systems use various schemes for communication, resource discovery, data locaion, application coupling, floor control and session control.

A prior research project involving the authors developed a LAN based collaboration architecture and toolkit called Shastra. The Shastra collaboration architecture[2, 3, 4] defines an extensible, programmable, computing environment for experimenting with sophisticated collaborative applications that support LAN based collaboration amongst multiple participants and systems. The architecture is composed of stream based collaboration aware network servers. Servers are classified into one of three categories: Fronts, which are the applications that users interact with; Session managers are specialized servers that control the interaction of Fronts; and Kernel's which act as global nameservers maintaining lists of available network resources. The architecture uses a hybrid data location scheme where network resource location information is replicated across all Kernels and session data is centrally located in Session Managers. The LAN based Shastra architecture is implemented in C++, X Windows and OpenInventor. A collection of toolkits and multimedia services have been developed to demonstrate the capabilities of the architecture. Sample toolkits are an Interactive Surface Modeling and Analysis tookit[15], Ganith a curve and surface manipulator, and Vistool a scientific visualization application. Multimedia services include Text, Audio, Video, and Whiteboard collaboration tools.

3 DSAV Loops for Collaborative Scientific Visualization

Collaborative scientific visualization can be loosely defined as a collection of visualization users who wish to share the results and control of their simulations and visualizations. These users may be located at geographically dispersed sites using different computational equipment and having access to widely varying hardware, resources, and data. The common bond amongst these users is that they wish to share their resources and results to aid each other in their research. DSAV loops are a model of collaborative work that provide users with an intuitive understanding of the activity taking place within a distributed simulation environment. This



(a) (b)

Figure 2: (a) presents an iso-surface of the hydraulic conductivity variable within a contaminated groundwater simulation. (b) shows a close up of an iso-surface of the same variable colored according to the concentration of contaminant dissolved in the groundwater.

model provides an easy to understand organization of distributed and cooperative tasks as well as a natural naming scheme and display for collaborative tasks, users, and activities.

A model of collaborative work needs to capture the available resources, the users involved in the work, the projects users are engaged in and the tasks needed to complete these projects. This model must not only provide users with this knowledge but must also provide an organization of it such that it allows users to discuss and manipulate the environment using this knowledge in an effective manner. For example existing client/server visualization tools allow shared views, providing users the ability to share the results of visualizations. However sharing of views is of limited utility in a WAN environment due to different work schedules of collaborators and potentially differing time zones. To fully support collaborative scientific visualization users must be able to share task knowledge in addition to view sharing.

The DSAV loop model organizes scientific visualization and simulation around a "loop" of project work. A work project in the simulation environment is viewed as a loop of activity with each basic component receiving information from the previous member of the loop and providing information to the next loop component. This basic loop is depicted in Figure 1(a). Components of the loop are data sources (D), simulation servers (S), analysis tools (A), and visualization clients (V). Its important to note that this organization of work does not preclude a server being used for multiple components of the loop. Thus a massive parallel system could fulfill all components of the loop but visualization. The loop model is a means of organizing, interacting with and planning shared projects and tasks. This loop nature of simulation and visualization work is well known in current systems but perhaps not clearly explicitly defined. In order to enhance a visualization and simulation environment to support collaborative activity we extend the DSAV loop into the third dimension, with multiple loops existing along the new axis. Using this three dimensional model of multi-loop interaction we can form connections across loops signifying shared tasks and projects. This shared loop model is depicted in Figure 1(b). The model provides a simple arrangement that allows for the rapid location, identification and manipulation of all shared resources within the collaborative system, and provides a description of the shared state of the collaborative environment. This description allows users to cooperatively arrange and manipulate shared information.

Users initiate new sim/viz projects by creating DSAV loops. Every DSAV loop created has a unique network wide identifier associated with it. This unique name allows the loop to be easily identified across all user sites. Loops maintain basic permissions as to who may add and remove components from the loop. Permissions associated with loops are the familiar owner, group, and other permission set. Initially a loop may only be modified by its creator/owner. Users can selectively add and remove components from a loop. All data generated by a component is associated with its current loop. The user interface for basic loop control can be seen in Figure 3.

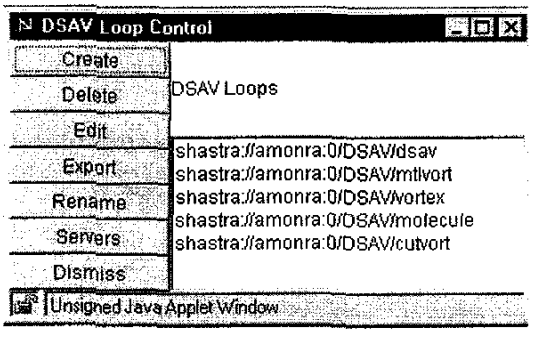


Figure 3: The user interface used to make basic manipulation of DSAV loops. The interface allows for creation, removal, and basic manipulation of loops. Users can share the user interface remotely throught the collaboration substrate.

Once a loop has been created the various project components can be added to the loop. Not all components need to exist within a loop. A shareable visualization session consists of at minimum a single DSAV loop. This loop may contain data sources, simulators, analysis tools, and visual-

izers. A network resource may act as more than one component. For example a massively parralel machine could act as a data source, simulator and an analyisis server. The division of resources into different components is a conceptual one and is done to provide users with a means of thinking about the activities in collaborative visualization. It is not an actual division of processes and server code. Each of the components in the model is represented as a shared network resource. Users add and remove components to individual loops using the interface shown in Figure 4. Accessible remote servers are available to users from a list of all servers maintained by the underlying distribtion substrate. Users may select and insert these servers into DSAV loops. The loops maintain the information specifying what role the component is fulfilling inside the project.

Servers that can act as components inside of a DSAV loop project must be able to take on one or more of the server roles described below:

Data sources maintain collections of data sets that can be directed to other servers for further computation. They must respond to basic requests for listing objects, assigning targets for data items, forwarding and retrieving data items. Data sources can be file systems, web servers, general servers, and scientific instruments.

Simulators are compute engines that take in data items and solve simulation problems such as fluid flow, and finite element analysis. For full integration into DSAV loops simulators should be fully instrumented to allow for computational steering.

Analysis servers handle results of simulators, and produce visualizble data sets. Analysis tools perform operations such as decimation, iso-surface and iso-contour extraction, and streamline generation.

Visualizers are used to control DSAV loops, visualize the results of components and steer simulations. They also provide collaborative support for multi-user system manipulation.

Two interface panels are provided for interacting directly with the servers that act as components for DSAV loops. These panels are provided as a means for the user to manage servers directly and to browse and control the state of the servers. These interfaces are shown in Figure 5.

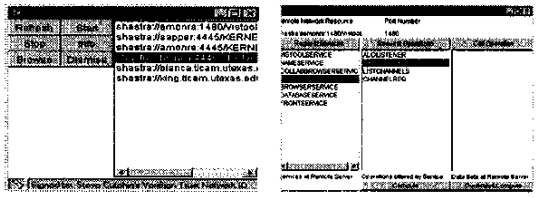


Figure 5: These panels provide users with direct control over system servers. They allow users to start,stop, browse and steer remote computation and analysis servers.

Users may specify connections between components in different loops. Connections between components of a DSAV loop are inter-loop connections. Connections between components of seperate DSAV loops are intra-loop connections. The visualizer tool controls the connections between loop components. The user interface in Figure 4 is used to specify connections between components. These connections create notions of common tasks. Inter-loop connections specify those tasks that will assist in the completion of the loop project. Intra-loop connections specify those tasks that will assist in the completion of an external project from the components DSAV loop. Visualizers can belong to multiple loops. The model also provides an easily displayable representation of user and task activity within a collaborative visualization. It can be displayed both as a 3d representation or as a hierarchy based list grouped around individual loops. Various different views can be selected, view of an individual loop, view of inter-loop connections, view of intra-loop connections, view of loop users, etc.

Developers implementing components within the the DSAV loop and cylinder model can use the connections between components to aid in decisions about pre-fetching data sets between servers. Servers connected within and between loops would have data automatically pre-fetch even while a server is still working on another task as the user has already specified a connection for the next task. The underlying collaboration substrate provides built in support for network identification and consistency management of shared objects (loops). The substrate also provides systemwide directory services providing applications and servers with full knowledge of all system resources, loops, components, and users.

The DSAV loop and cylinder model gives users a mental picture of the connection of distributed components and the work being done. Loops create in the users mind a notion of a common project. A visual representation of the model provides users with organized collaboration awareness of other users and shared tasks.

This interaction model has many benefits for distributed visualization of simulations. Use of the model changes the nature of the work from data and server management to the collaborative management of shared project tasks without regard to user location or schedule. It provides a framework for easily distributing tasks over a pool of network resources, increasing the available resources in terms of both computers, software and personnel. The model creates an environment that supports shared task management and provides collaborative task awareness. The use of the Shastra substrate in conjunction with the model provides flexible coupling to support user determined division of labor for working on simulation visualization and control. The model enhances the ability to arrange multiple views of the same data set. Synchronous and asynchronous work is supported by the underlying collaboration substrate. The model organizes tasks into such a framework that tasks have meaning over the lifetime of a project without direct explanation by remote users.

4 Implementation of Model

The DSAV interaction model was implemented using the Shastra collaboration toolkit. A set of WAN specific distribution and collaboration features was used to support effective collaboration over the Internet. A re-usable visualization module implemented as a downloadable Java applet provides web access from any Java enabled web browser. The visualization applet acts as the generic user interface into the DSAV loop control evenvironment.

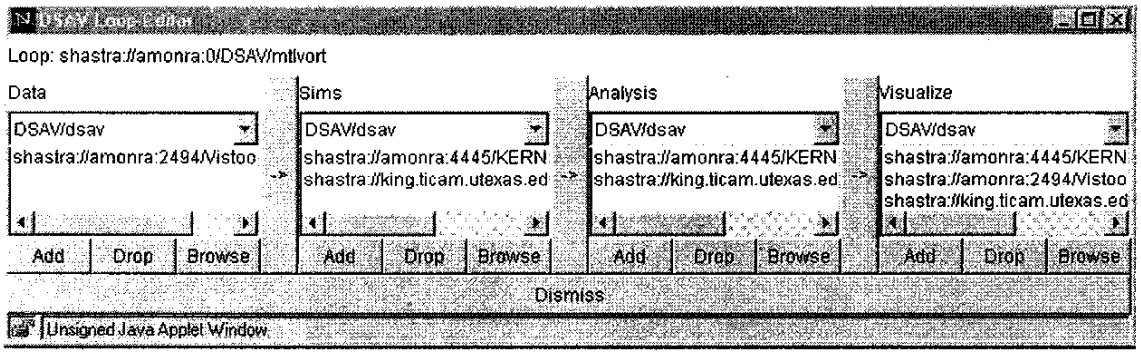


Figure 4: This interface allows users to add components as Data, Simulation, Analysis, and Visualizer, to a DSAV loop. They can from a list of available servers, or from any existing DSAV loop. The panel allows connections to be formed by selecting servers and simply clicking on the arrow button. This creates a connection between servers.

4.1 Collaboration Substrate

The Shastra Collaboration Architecture[2, 3, 4] defines an extensible, programmable, computing environment for developing, using, and experimenting with sophisticated collaborative applications that support large scale geographically dispersed collaboration amongst multiple participants and systems. The architecture is connectionless, scalable, supports multi-group federation, has an adaptable data location model, supports flexible application coupling, multiple coordination strategies, dynamic downloading of executable code and is event driven. The architecture defines a collection of specialized servers providing server specific services which communicate using network events. The interaction of these specialized servers creates an environment capable of supporting collaborative work. A collection of specialized servers and services has been created that provide collaboration awareness, data management, and coordination control to multiple users. The system is available in both Java and C++ versions. A pictorial representation of the architecture is given in Figure 6.

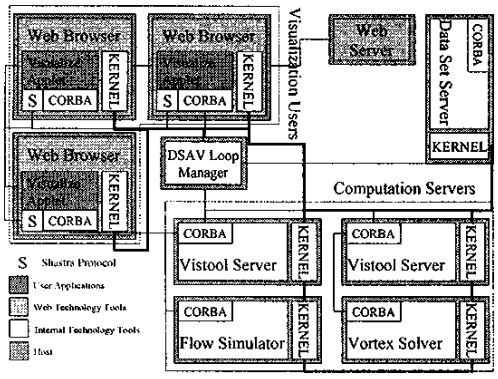


Figure 6: The runtime architecture of the DSAV environment implemented with the Shastra substrate. It shows three user browsers(upper left), four computation engines(lower right), and a web server and data source (upper right).

4.2 New WAN Collaboration Features

The new Shastra architecture provides several WAN specific features that significantly aid the creation of Internet based collaboration tools. A mechanism for supporting shared objects using optimistic consistency is used for managing the distributed DSAV loops. The technique, called Operation Transformation[11, 14], allows shared objects to truly be simultaneously and asynchronously manipulated without locking while maintaing the objects in a consistent state. Figure 7 depicts the architecture of the system.

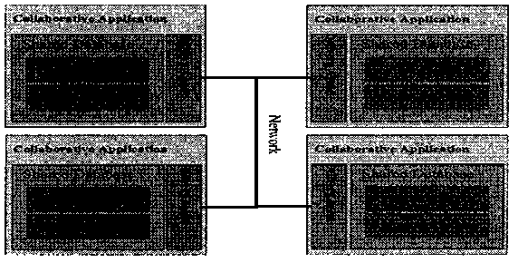


Figure 7: The architecture of the shared object mechanism using operation transformations.

An adaptive data location algorithm is used to adjust the distribution of shared objects over the network to improve communication and enhance overall system performance. The data location algorithm dynamically adjusts the location of objects and the transmission of information based on communication cost, network activity, computation cost, and network topology. The system uses dynamically created routing agents that cache objects and operations and adjust their location and storage based upon network activity. The architecture of the adaption mechanism is depicted in Figure 8.

A collaborative widget set that uses operation transformation to maintain consistency amongst distributed user interfaces is used to provide shared multi-user interfaces. These multi-user interfaces are used by collaborating users to manipulate DSAV loops as well as interact with shared views, and shared geometry objects. Coupling filters which enhance the capabilities of these shared widgets by providing the user with full control over the degree of connection between shared interfaces are used to enhance multi-user

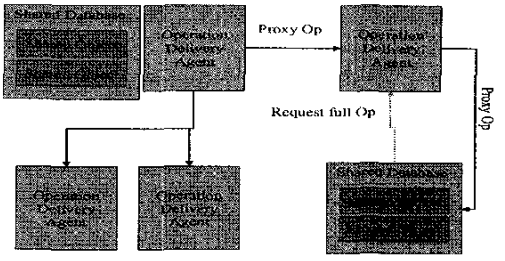


Figure 8: The basic communication route of operations through a delivery agent. Delivery agents are used to adaptively route operation transmission according to vital network parameters.

flexiblity. These filters allow users to selectively control what components of a user interface are shared between collaborating applications. The basic architecture is depicted in Figure 9.

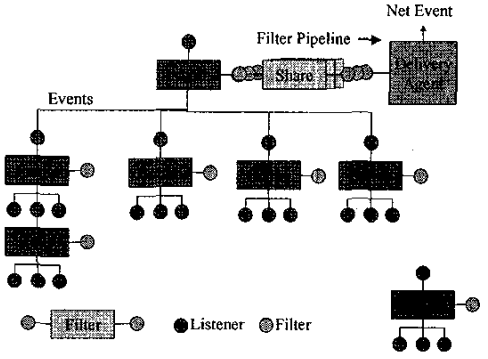


Figure 9: This figure depicts the hierarchical event driven organization of the shared widget set. Widgets in the system generate events/operations that are transmitted to the user interfaces of all collaborating users. The filters in the graph allow for users to control the degree of coupling betweeen their interfaces.

4.3 Collaborative Visualization Module

A modularized visualizer that supports collaborative view sharing, distributed resource access and compressed VRML display is used as the general visualization tool. The visualizer is a pure Java client capable of displaying VRML scenes, including compressed geometry VRML scenes and binary format VRML scenes. It supports collaborative navigation and interogation of shared scene graphs. The visualizer can be used independently or collaboratively to view objects and scenes retrieved from Web servers, JDBC compliant databases, and the DSAV environment. The visualizer is available as a Web based Applet using the VRML External Authoring Interface. This allows it to execute in any Java enabled web browser with a VRML plugin. This ensures the widest possible distribution and access to the visualizer. Currently versions running in Netscape Communicator and Microsoft Internet Explorer are available. The visualizer is also available as a standalone Java application using the Java3D VRML browser and supporting CORBA interaction.

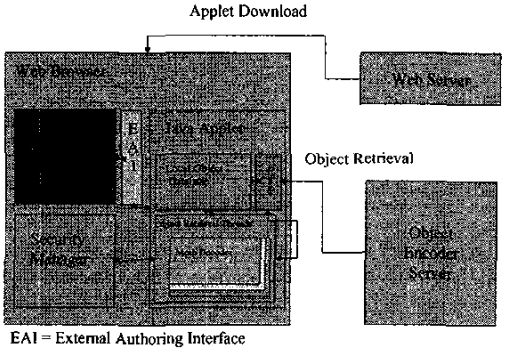


Figure 10: The figure depicts that static architecture of the visualization applet. The applet is composed of java and VRML components. The applet maintains a set of objects and streams in the object database and uses the security manager to extend privileges needed for advanced network access.

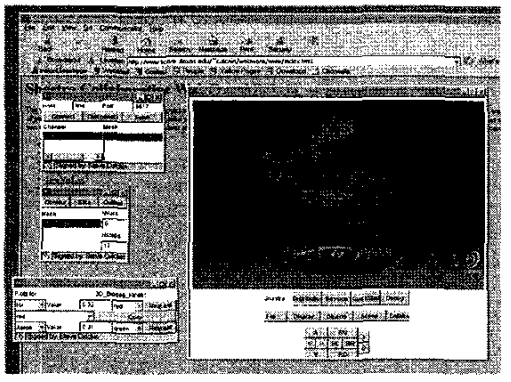


Figure 11: This is a snapshot of the visualizer and control panels used to manage and manipulate plots, simulators, and analysis tools. The panels on the left control access to servers, plot display, and mesh manipulation. The panel on the right displays the visualization data and provides navigation control.

5 3D Soil Contaminant Remediation Simulation and Visualization

An application has been developed that demonstrates the utility of the DSAV loop interaction model. The Center for Computational Visualization along with the Center for Subsurface Modeling at the University of Texas at Austin are using the model to to understand the complex processes by which contaminated groundwater can be remediated in situ. In Figure 2(b), plumes of dissolved contaminant flow

downstream from pools of immobile nonaqueous phase liquid (NAPL). The concentration of contaminant varies with position and time due to kinetic competition between the dissolution rate, the rate of biodegradation by naturally occurring bacteria, the consumption of oxygen by the bacteria and the consumption of oxygen by reducing minerals found in the aquifer. A set of tools and servers have been developed and/or modified to support DSAV loops and allow users to experiment with simulations in contaminant remidation through the world wide web.

The application uses the following tools and servers:

data source a Web server is used as the source of the initial data set for simulation and analysis.

simulator A flow solver called PARSSIM developed at the Universtiy of Texas Center for Subsurface Model is used to solve the flow problems describing the soil conductivity. The current version runs on multiple platforms including the IBM SP2.

analysis A server name Vistool is used to perform visualization analysis. The tool supports decimation, isocontour and iso-surface extraction, slicing, and mesh refinement.

visualizer The Java web visualizatio applet is used as the visualizer for displaying and control the servers, data, and tools.

Figure 2 shows sample screenshots of the visualizations of the results of the simulation. The user interacts with the various components of the DSAV loop through the visualization applet. The visualization applet provides the user-interface for creating shareable DSAV loop's. Within the visualizer the user can select and specify flow solvers and vistool servers to include within specific DSAV loops. The current incarnation of the PARSSIM flow solver provides simple start/stop and job submission controls. Work is being done to enhance it and allow the user to modify simulation parameters over the course of the computation of a flow solution.

6 Example Scenario

A simple testing scenario was performed using facilities at the University of Texas at Austin. Three researchers on different machines engaged in collaborative visualization and simulation experimentation with the developed system to explore usability and performance. The hardware and software configuration was as follows: One user was on WindowsNT using Internet Explorer 4.0 with the Cosmoplayer plugin for visualization. The second user also ran on WindowsNT but used Netscape Communicator 4.5 and the Cosmoplayer VRML plugin. Finally the the third user was using Netscape Communicator on a Silicon Graphics Onyx2 system, with Cosmoplayer plugin. Simulation and analysis servers ran on an IBM SP2 and on a SGI Onyx2 respectively. Three sample snapshots from the experiment are display in Figure 12.

This research is supported in part by NSF grants CCR-9732306 and KDI-DMS-9873326

References

[1] Mark Abbott and Lalit Kumar Jain. DOVE: Distributed Objects Based Scientific Visualization Environment. In ACM 1998 Workshop on Java for High-Performance Network Computing . ACM Press, 1998.

[2] V. Anupam and C. Bajaj. Collaborative Multimedia Scientific Design in SHASTRA. IEEE Multimedia, pages 39–49, 1994.

[3] V. Anupam and C. Bajaj. SHASTRA - An Architecture for Development of Collaborative Applications. International Journal of Intelligent and Cooperative Information Systems, pages 155–172, 1994.

[4] V. Anupam, C. Bajaj, F. Bernardini, S. Cutchin, J. Chen, D. Schikore S. Evans, G. Xu, and P. Zhang. Scientific Problem Solving in a Distributed and Collaborative Environment. Mathematics and Computers in Simulation, 36:433 – 542, 1994.

[5] A. Baratloo, M. Karaul, H. Karl, and Z. Kedem. KnittingFactory: An Infrastructure for Distributed Web Applications. Technical Report Technical Report TR 1997-748, New York University, November 1997.

[6] Lukasz Beca, Gang Cheng, Geoffrey C. Fox, Tomasz Jurga, Konrad Olszewski, Marek Podgorny, Piotr Sokolowski, Tomasz Stachowiak, and Krzysztof Walczak. Tango - a collaborative environment for the world-wide web. Technical report, Northeast Parallel Architectures Center, Syracuse University.

[7] R. Bentley, T. Horstmann, K. Sikkel, and J. Trevor. Supporting Collaborative Information Sharing with the World Wide Web: The BSCW Shared Workspace System. In *Proceedings of the Fifth Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 1996.

[8] Michael Boyles, Rajeev Raje, and Shiaofen Fang. CEV: Collaborative Environment for Visualization Using Java RMI. In ACM 1998 Workshop on Java for High-Performance Network Computing . ACM Press, 1998.

[9] Annie Chabert, Ed Grossman, Larry S. Jackson, Stephen R. Pietrowiz, and Chris Seguin. Java object-sharing in Habanero. *Communications of the ACM*, 41(6):69–76, June 1998.

[10] K. Mani Chandy, Adam Rifkin, Paolo A.G. Sivilotti, Jacob Mandelson, Matthew Richardson, Wesley Tanaka, and Luke Weisman. A world-wide distributed system using java and the internet. In *IEEE International Symposium on High Performance Distributed Computing (HPDC-5)*, August 1996.

[11] Steve Cutchin. *Web Based Collaboration Techniques*. PhD thesis, Purdue University, August 1999.

[12] S. Dossick and G. Kaiser. WWW Access to Legacy Client/Server Applicatons. Technical Report CUCS-003-96, Columbia University, 1996.

[13] Greg Eisenhauer, Weiming Gu, Eileen Kraemer, Karsten Schwan, and John Stasko. Online Displays of Parallel Programs: Problems and Solutions. In

Hamid R. Arabnia, C. Ierotheou, Ronald A. Olsson, Yi Pan, R. Pandrey, and E. G. Talbi, editors, *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, PDPTA'97*, pages 11–20, Las Vegas, NV, July 1997.

[14] C. A. Ellis and Simon J. Gibbs. Concurrency control in groupware systems. In *ACM SIGMOD International Conference on the Management of Data*, pages 399–407, 1989.

[15] Susan Evans. *Interactive Surface Modeling and Analysis.* PhD thesis, Purdue University, August 1997.

[16] T. Frivold, R. Lang, and M. Fong. Extending WWW for Synchronous Collaboration. In *Proceedings of the Second World Wide Web Conference'94: Mosaic and the Web*, 1994.

[17] Yves Jean, Thomas Kindler, William Ribarsky, Weiming Gu, Gregory Eisenhauer, Karsten Schwan, and Fred Alyea. Case study: An integrated approach for steering, visualization, and analysis of atmospheric simulations. Technical Report 95-15, Georgia Institute of Technology. Graphics, Visualization and Usability Center.

[18] Arie Kaufman. *Volume Visualization.* IEEE Computer Society Press, 1991.

[19] B. Kvande. The Java Collaborator Toolset, a Collaborator Platform for the Java(tm) Environment. Master's thesis, Old Dominion University, August 1996. master's thesis.

[20] Maaretn and Sierhuis. Towards a framework for collaborative modeling and simulation. In *Proceedings of CSCW 96*, 1996.

[21] J. S. Vetter. Computational Steering Annotated Bibliography. Technical Report GIT-CC-94-15, Georgia Tech, May 1997.

[22] Jason D. Wood, Helen Wright, and Ken W. Brodlie. Collaborative visualization. In Roni Yagel and Hans Hagen, editors, *IEEE Visualization 97*, pages 253–260. IEEE, November 1997.

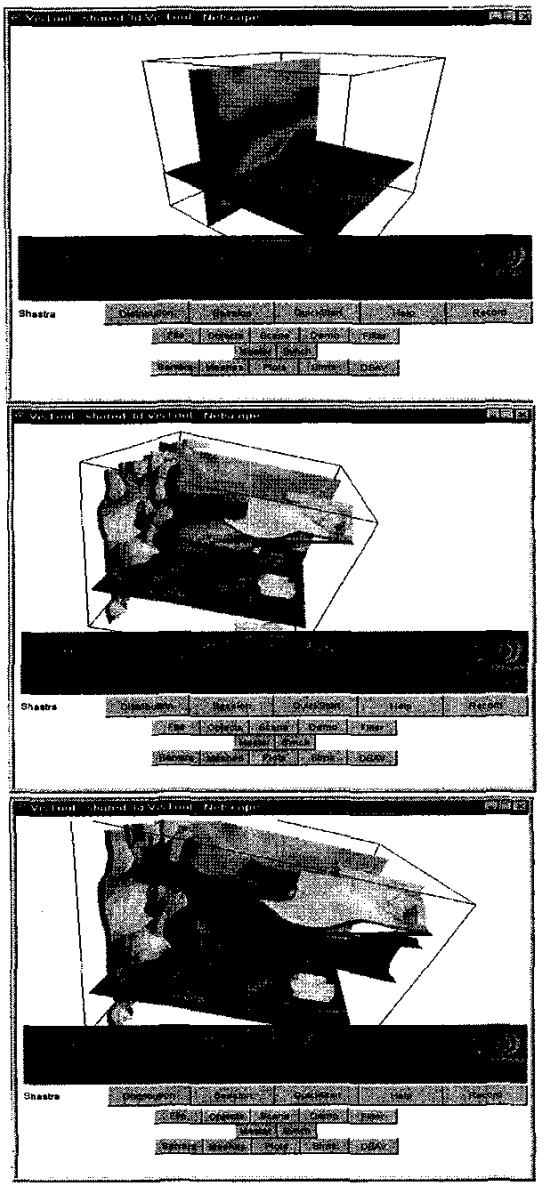


Figure 12: These are sample snapshots taken from three different users visualizations applets on three different machines. The first picture shows slices from the visualization, the second hydraulic conductivity of the soil, and the third a NAPL.