Computational Science and Engineering Online (CSE-Online): A Cyber-Infrastructure for Scientific Computing

Thanh N. Truong,*,† Manohar Nayak,† Hung H. Huynh,† Tom Cook,† Priya Mahajan,† LeThuy T. Tran,† Jannu Bharath,† Shrish Jain,† Ha B. Pham,† Chaiwoot Boonyasiriwat,† Nhat Nguyen,† Evan Andersen,† Yong Kim,† Suengkeol Choe,† Jihoon Choi,† Thomas E. Cheatham III,‡ and Julio C. Facelli§

Department of Chemistry and the Interdisciplinary Computational, Engineering, and Science Program, Departments of Medicinal Chemistry and of Pharmaceutics and Pharmaceutical Chemistry and of Bioengineering, and Center for High Performance Computing, University of Utah, 315 South 1400 East, Rm 2020, Salt Lake City, Utah 84112

Received September 9, 2005

With the expansion of the Internet and World Wide Web (or the Web), research environments have changed dramatically. As a result, the need to be able to efficiently and securely access information and resources from remote computer systems is becoming even more critical. This paper describes the development of an extendable integrated Web-accessible simulation environment for computational science and engineering called Computational Science and Engineering Online (CSE-Online; http://cse-online.net). CSE-Online is based on a unique client−server software architecture that can distribute the workload between the client and server computers in such a way as to minimize the communication between the client and server, thus making the environment less-sensitive to network instability. Furthermore, the new software architecture allows the user to access data and resources on one or more remote servers as well as on the computing grid while having the full capability of the Web-services collaborative environment. It can be accessed anytime and anywhere from a Web browser connected to the network by either a wired or wireless connection. It has different modes of operations to support different working environments and styles. CSE-Online is evolving into middleware that can provide a framework for accessing and managing remote data and resources including the computing grid for any domain, not necessarily just within computational science and engineering.

1. INTRODUCTION

> ...a new age has dawned in scientific and engineering research, pushed by continuing progress in computing, information, and communication technology, and pulled by the expanding complexity, scope, and scale of today's challenges. The capacity of this technology has crossed thresholds that now make possible a comprehensive "cyberinfrastructure" on which to build new types of scientific and engineering knowledge environments and organizations and to pursue research in new ways and with increased efficacy.
>
> —NSF Blue Ribbon Advisory Panel report on cyberinfrastructrure, 2003[1]

There have been a number of activities directed toward achieving the vision outlined above, such as the creation of "collaboratories", computing "grids", e-science, and numerous domain-specific projects aiming to create a new cyberinfrastructure. Such projects would ultimately induce a paradigm shift in the way we do research, access scientific data, and collaborate.

All of these activities recognize the fact that, with the expansion of the Internet and World Wide Web (or the Web), research environments have changed dramatically. As a result, the need to be able to efficiently and securely access information and resources from remote computer systems is becoming even more critical. Furthermore, in many cases, and particularly for computing-intensive work, a remote system is not just one computer or server but rather a collection of computers connected over the Internet or via a grid structure, such as the TeraGrid.[2] The same is also true for data-intensive work where data can be stored in distributed databases located on many computers at different locations.

Nearly all computers for scientific computing purposes are running Unix-based operating systems. A typical work environment for a user to access these computers remotely is to open an X-window session on their local computer to connect directly to these servers using the secure shell (SSH) protocol.[3] The user may choose to connect to more than one server concurrently. In this case, the user has an account, owns directories on each of these servers, and communicates directly with the server operating system (OS) using command line commands. Users do not see the desktop environment of the server using SSH. To visualize data, users can run an application that exports a display to the local computer from the server using the X11 forwarding protocol.[4] The X11

* Corresponding author e-mail: truong@chem.utah.edu.
† Department of Chemistry and the Interdisciplinary Computational, Engineering, and Science Program.
‡ Departments of Medicinal Chemistry, Pharmaceutics and Pharmaceutical Chemistry, and Bioengineering.
§ Center for High Performance Computing.

10.1021/ci0503917 CCC: $33.50
Published on Web 02/21/2006

protocol can "serve out" the screen, keyboard, mouse, and so forth. Unfortunately, the X11 protocol often has difficulties in exporting graphics from the server. In this technology, all events are executed on the server. Stability can be a problem because a weak link in this system is the network connection. Because the network connection is used constantly, the entire X11 session fails when the network connection fails, and it requires significant network bandwidth for real-time performance.

Recent advances in Web-services technologies[5] have led to a different mechanism for accessing remote data and resources. The possibility of using such Web-services technologies to provide a new framework for scientific computing also attracted a lot of interest in the computer science community in the mid-1990s. Numerous reports illustrate the potential of a Web-based framework for simulations.[6−10] However, realization of such potential was rather limited, and the interest in this area has significantly dropped since 2000 due to the lack of real applications as pointed out by Kuljus and Paul.[10] Fortunately, interests in the development of Web-based simulation environments have been revitalized recently by domain scientists, particularly in conjunction with the computing Grid. For instance, within the molecular science simulation community, there are a number of projects, such as the development of Collaborative Multi-Scale Chemical Science (CMCS),[11] Grid-Enabled Molecular Science Through Online Networks Environments (GEMSTONE),[12] Computational Chemistry Grid (GridChem),[13] Discovery Net,[14] Comb*e*Chem,[15] Reality Grid,[16] and the Virtual Laboratory Project,[17] among others.

All of these Web-based collaborative or simulation environments share the common Web-services software architecture. With these technologies, the users access information and request services provided by a third-party provider using a nonsecure (http) or secure (https) connection. The Web application can perform a variety of tasks ranging from the simple delivery of data to complicated processes that involve the authentication and execution of remote processes. In the Web-services architecture, users have a relatively thin client that communicates directly with the Web-service provider and not with the remote server OS. Thus, the authentication process is performed by the remote server application and not by the server's OS. In addition, users do not generally own accounts and directories on the remote server. The Web application may provide Web accounts and virtual directories for the end user, but this is significantly different from having an account directly on the server itself. For instance, files in virtual directories are not owned by the user. Requests for resources on the server are done by the server application and not directly by the user. This environment is quite useful for those who only care that the requested data or service, that is, the end result, is delivered and do not care how the data was stored or about the process for generating such data. This environment is very different from opening an SSH session on the remote computer to access the user's own data and applications.

The largest impact of Web-services technology within science and engineering recently has been in the development of collaborative environments called "collaboratories", where multiple users can share data, applications, and instruments as well as communicate with each other in solving specific complex scientific problems.[18−21] These collaboratories, for the most part, can be classified into two types: data-sharing-oriented or remote-access scientific-instrument-oriented. The Research Collaboratory for Structural Bioinformatics[22] is an excellent example of a data-oriented collaboratory that provides access to databases of biological structures and tools for determining and analyzing these structures. Most existing collaboratories are instrument-driven and provide the capability for real-time data acquisition from remote research instruments through Web-accessible servers in a seemingly transparent way and are often focused specifically on a particular complex scientific problem. The Space Physics and Aeronomy Research Collaboratory[23] provides an Internet-based collaborative environment for studies of space and upper-atmospheric science facilitating real-time data acquisitions from a remote site in Greenland. The Materials Microcharacterization Collaboratory[24] provides remote access to facilities that perform electron-beam microcharacterization of materials. The Environmental Molecular Science Collaboratory[25] allows remote instrument control of the NMR and FTICR spectrometers located at the Pacific Northwest National Laboratory.

In summary, these two technologies complement each other, that is, SSH for individual computing work and Web-services for collaborative work. To the best of our knowledge, no technology currently exists that can support both mechanisms in the same framework. Thus, the current paradigm for a computing work environment requires the individual to *manually* log on to these servers *separately* to access the data and applications with technologies that are inefficient and cumbersome to use, require a steep learning curve, are unstable with regard to the Internet interruptions, and often require a large bandwidth for graphics applications, while collaborating the results with others is difficult because existing collaborative environments are based on a different and disjointed software framework.

In this paper, we describe our current efforts in developing an extendable integrated Web-accessible simulation environment for computational science and engineering, called Computational Science and Engineering Online (CSE-Online).[26−28] CSE-Online introduces a new paradigm by allowing diverse types of application tools, data, and computing resources distributed over the Internet on different servers to be accessed *concurrently* from a *single* virtual desktop environment in a *simple*, *secure*, and *transparent* manner. The innovation of CSE-Online is in its ability to allow both individual and collaborative work environments in the same software framework. This paper is organized as follows. Section 2 describes the overall software architecture of CSE-Online and its implementation. The following sections provide further details related to its four major components, namely, the knowledge management system, visualization/analysis, collaborative environment/communication, and computing environment components, respectively. In the discussion of these components, our focus is on the high-level design and functionalities. We also provide some discussion on the technologies used for implementation.

2. SOFTWARE ARCHITECTURE AND IMPLEMENTATION

The main goal of this integrated, extendable, platform-independent, and secured environment is that it enables a

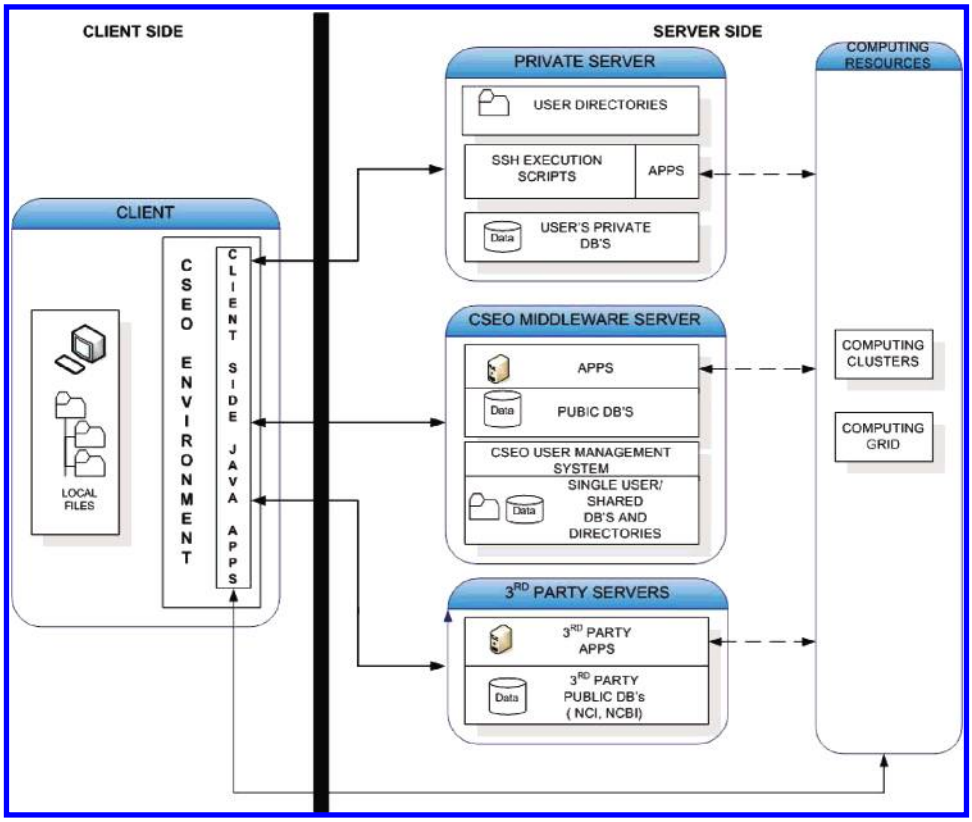

Figure 1. Plot of the overall software architecture of CSE-Online.

seamless interface and data flow between a variety of different computational application tools and databases. This environment pulls together independent application tools and databases to provide a complete bottom-up environment that integrates methods ranging from quantum chemistry to reaction engineering, or to computer-aided drug design, which also involves homology modeling, molecular docking, molecular dynamics, and free energy simulations while having the look and feel of a single desktop application. It provides a knowledge management system that allows a user to store, query, retrieve, and mine data extracted by running these application tools. The environment also provides communication tools, which work in both synchronous and asynchronous modes, to facilitate training, user support, and collaboration. It can be accessed anytime and anywhere from a Web browser connected to the network by either a wired or wireless connection while still enabling visualization/analysis of remote data in real time.

Because of the complexity inherent in the specifications of the environment and the existence of numerous relevant technologies, to help us in the design of the software architecture as well as the technology choice, we rely on the following four guiding principles: (1) Having a good human experience at the interface is equally as important as solid software architecture. (2) It is critical to strike a balance between employing emerging technologies and providing users with stable and reliable tools. (3) It is important to recognize and support the common needs across disciplines, while providing a flexible framework that can be customized for each particular scientific domain. (4) Not only should individual research be empowered but collaboration across disciplines should be facilitated.

In addition, from a software engineering point of view, the architecture of this environment should also be scalable and extendable to ensure its future growth and support of new and different applications or databases.

Overall Software Architecture. CSE-Online is based on a client−server architecture model. The client CSE-Online environment consists of numerous client-side Java applications that allow the user to access and analyze remote data or use remote resources. It is discussed separately below. The CSE-Online server side has four different, separate components, and their connectivity to the client environment is different depending on their roles, as shown in Figure 1.

The central server component is the CSE-Online Middleware server, which is owned by CSE-Online. It hosts the CSE-Online Web site where users first log in from a Web browser to enable the CSE-Online executable file. It has a user management system, which allows the user to share files or databases with other CSE-Online users and contains the user preference information. It also contains applications and public databases provided by CSE-Online. The user accesses data or applications on the CSE-Online server using the Enterprise Java Bean (EJB) technology.[29]

The second component is the private server(s) (it can be more than one remote server). Private servers, in the context of this paper, means the servers where the user has direct access and prefers to store data or has access to application tools. CSE-Online has no direct access to these servers. These "private" servers may or may not be owned by the user. The user accesses the private server from the CSE-Online client environment using the SSH technology, which provides strong authentication and secure communications over insecure channels. This server can also contain private databases within the CSE-Online knowledge management system, as discussed in a later section. Some installation is required on this server to use this feature.

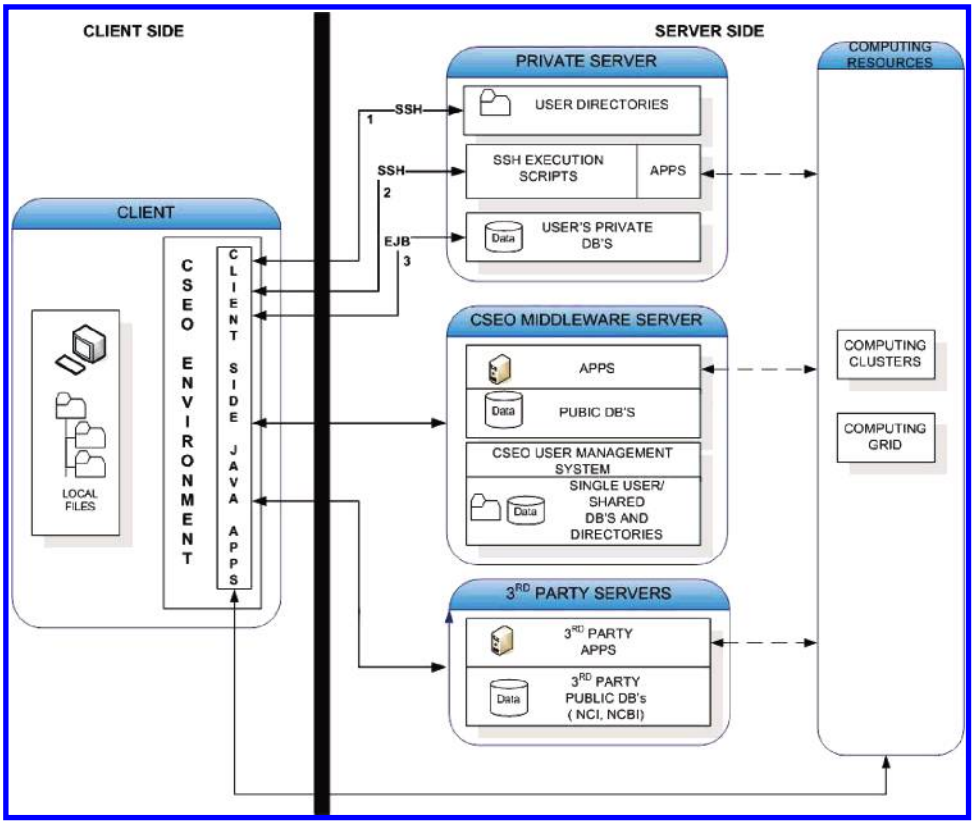

Figure 2. Plot to show the active components of the "connection to a private server" operational mode.

The third component is the third-party server. It consists of all databases and applications provided by the third party. Both CSE-Online and the user have no direct access to these servers (i.e., they have no directory accounts on these servers). Accessing databases or services from these third-party servers is done by using the Web-services[5] or http technology.[30] CSE-Online helps the user communicate with these Web services without the user needing to leave the environment.

The fourth component is the computing resources, which can be a remote cluster, distributed clusters, or even a computing grid such as the TeraGrid,[2] where the user has allocated resources. The user can access these computing resources from the CSE-Online environment by making a direct connection to these computing nodes using the SSH technology. They can also elect to request such resources from the CSE-Online client environment without making a direct connection to these computing nodes by using the Web-services technology if the computing nodes support such a service.

With the CSE-Online client environment and the four server components, this software architecture offers the user three distinctive modes of operation. Specifically, the three modes are (1) connection to a private server mode, (2) without connection to a private server mode, and (3) offline mode. The differences in these modes are mainly how CSE-Online accesses data and computing resources.

1. Connection to a Private Server Mode. Figure 2 illustrates the active components of the "connection to a private server" mode. In this mode of operation, the user wants their data to be on their private server(s) or the ability to access certain applications on this server. The File Manager, a component of the client environment described below, maps the user file directories on that server to the environment and, if enabled, allows the user to run any application on that server. The user can set up to have the ability to create CSE-Online private databases on this server. This system is ideal for users who (1) already have licenses for their applications, (2) want maximum security for their data, and (3) would like to maintain their own servers. The environment is stable with regard to network interruptions. Communication to the private server is done only during the read/write request. All visualization or analysis of data is done on the client side for maximum performance. A unique feature of CSE-Online is that the environment remains operational when the Internet is disconnected; that is, the client environment loses its connection to the private server. When a cache process is used, most data can be saved during such an event. Attempts to reconnect to the private server are done automatically. In addition, from the CSE-Online client environment, the user can also access data and applications provided by the CSE-Online server or third-party servers and remote computing resources including the computing grid.

2. Without Connection to a Private Server Mode. Figure 3 shows the active components of the "without connection to a private server" mode of operation. In this mode, the user prefers to have their data on their local computer rather than on a remote "private" server or does not own a private server. In such cases, the File Manager maps all local storage devices on the user's computer such as the local hard disk, CD-ROM, floppy, USB memory stick, and so forth. The user can access data and applications provided by the CSE-Online Middleware and third-party servers. They can also access the remote computing resources from the CSE-Online environment. In summary, the only difference to the user

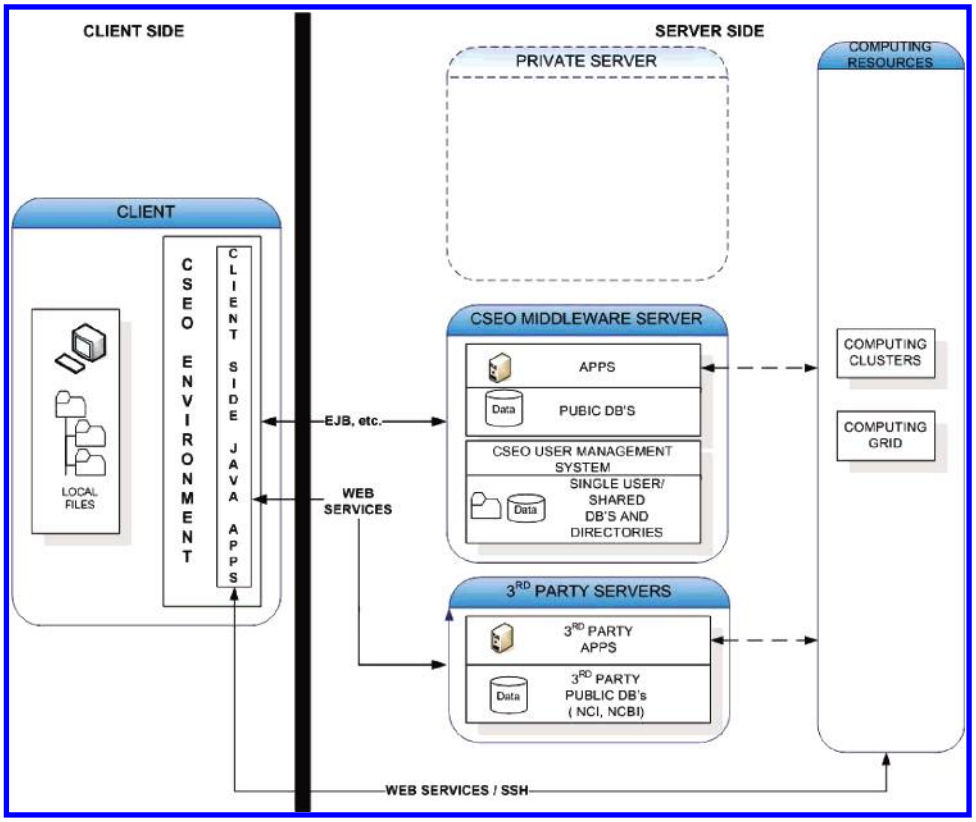

Figure 3. Plot to show the active components of the "without connection to a private server" operational mode.

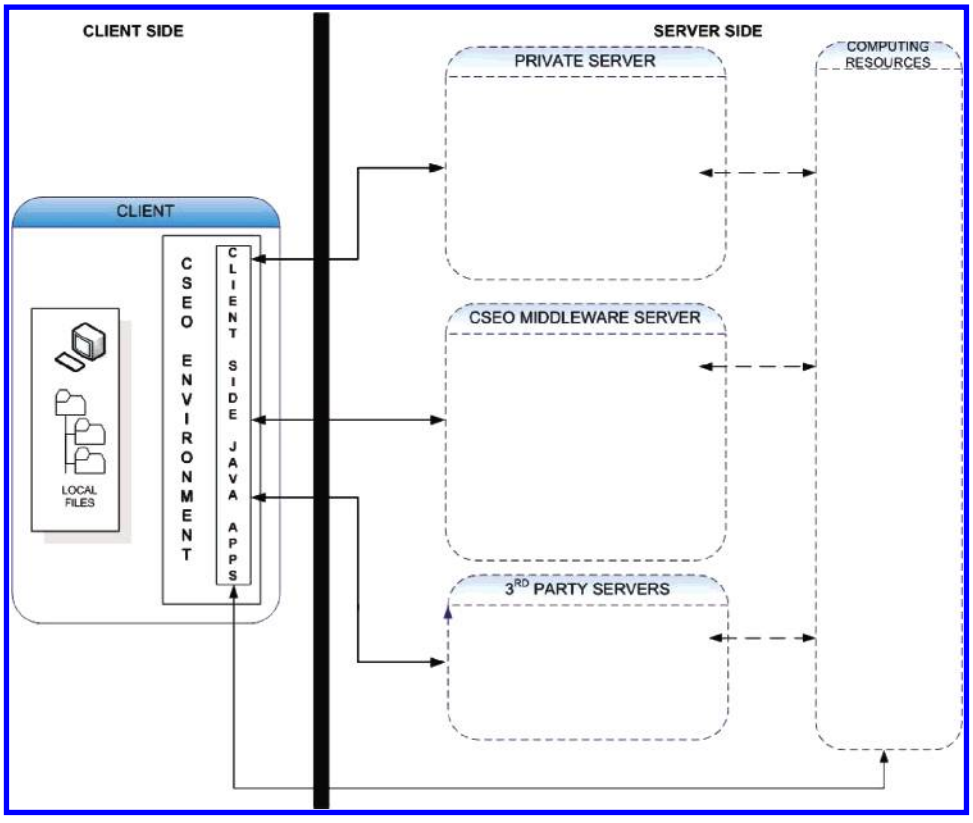

Figure 4. Plot to show that only the client environment is the active component in the offline mode.

between the "connection to a private server" and "without connection to a private server" modes is that there is no the private server component in the latter mode.

3. Offline Mode. Figure 4 shows only the client desktop environment to be the active component in the offline mode of operation. In this mode, the Internet connection is not available. The client environment behaves as a single integrated application operating on the local computer that can read/write data to the local storage devices. Since the CSE-Online client environment provides many Java-based application tools, such as data analysis and visualization, and also provides a "sync" capability for data on a private server

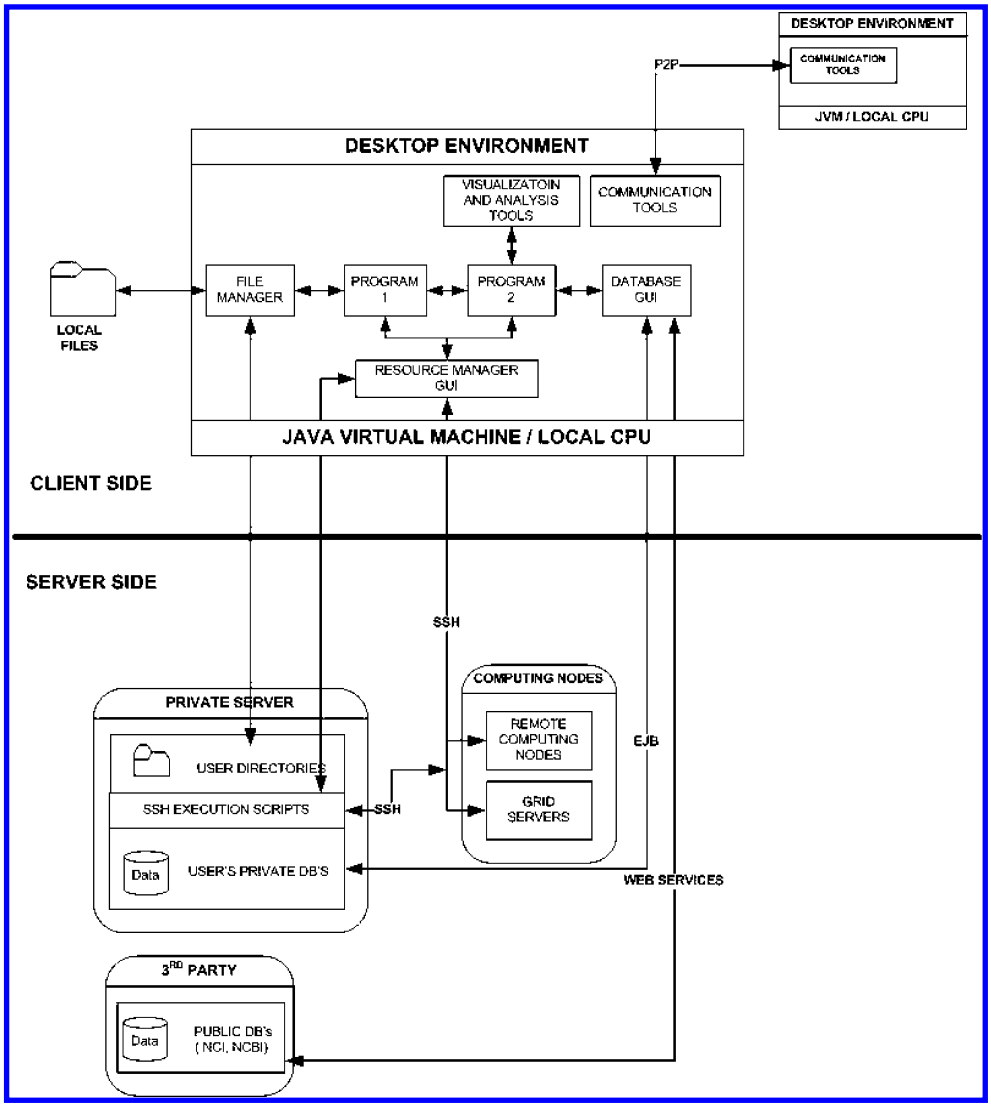

Figure 5. Plot of the software architecture of the user desktop environment. Different components of the environment and their connections to the server components are also shown.

and the user's local computer, the user can still be productive in the offline mode.

In the official released version, in May of 2005, of CSE-Online at http://cse-online.net, only the "connection to a private server" mode is available. The other two modes of operation are either being implemented or are at the α testing stage.

Desktop Environment. The CSE-Online client desktop environment is based on Java and Java Webstart technologies,[31] and thus, it allows the environment to be independent of the operating systems and Web browsers. In other words, its end-user look and feel is the same on all systems. Furthermore, it is a "thin client" environment; that is, no installation is needed by the user beyond the Java Runtime Environment (JRE) needed for running any Java applications. The executable of the desktop environment is downloaded to the user's local computer for the first time upon login onto the CSE-Online Middleware server. Subsequent uses on the same computer can access the environment executable from the local cache. It is important to point out that the CSE-Online desktop environment is based on an open software architecture. Such open architecture allows each tool in the desktop environment to be developed independently and, thus, provides the possibility for third parties to develop and integrate their own tools into the desktop environment via our published application programming interfaces (API). More details on APIs for each application within the CSE-Online environment will be available online from the CSE-Online site. Furthermore, the desktop application tools are loaded at run time, and thus, it allows the user to select only desktop tools that are needed for their work to be loaded into the environment. This allows the development of the desktop environment to be extendable; that is, the environment can support an unlimited number of desktop application tools. The size of the executable only depends on the number of tools that were selected by the user. The CSE-Online desktop environment resembles a standard Windows or Linux desktop environment with a toolbar listing application and utility tools that were selected by the user. The CSE-Online desktop environment has four major components, namely, the knowledge management system, resource management system, visualization/analysis, and communication components, as shown in Figure 5. These components are discussed separately below. In addition to these components, the CSE-Online desktop environment has a number of desktop application tools which can be separated

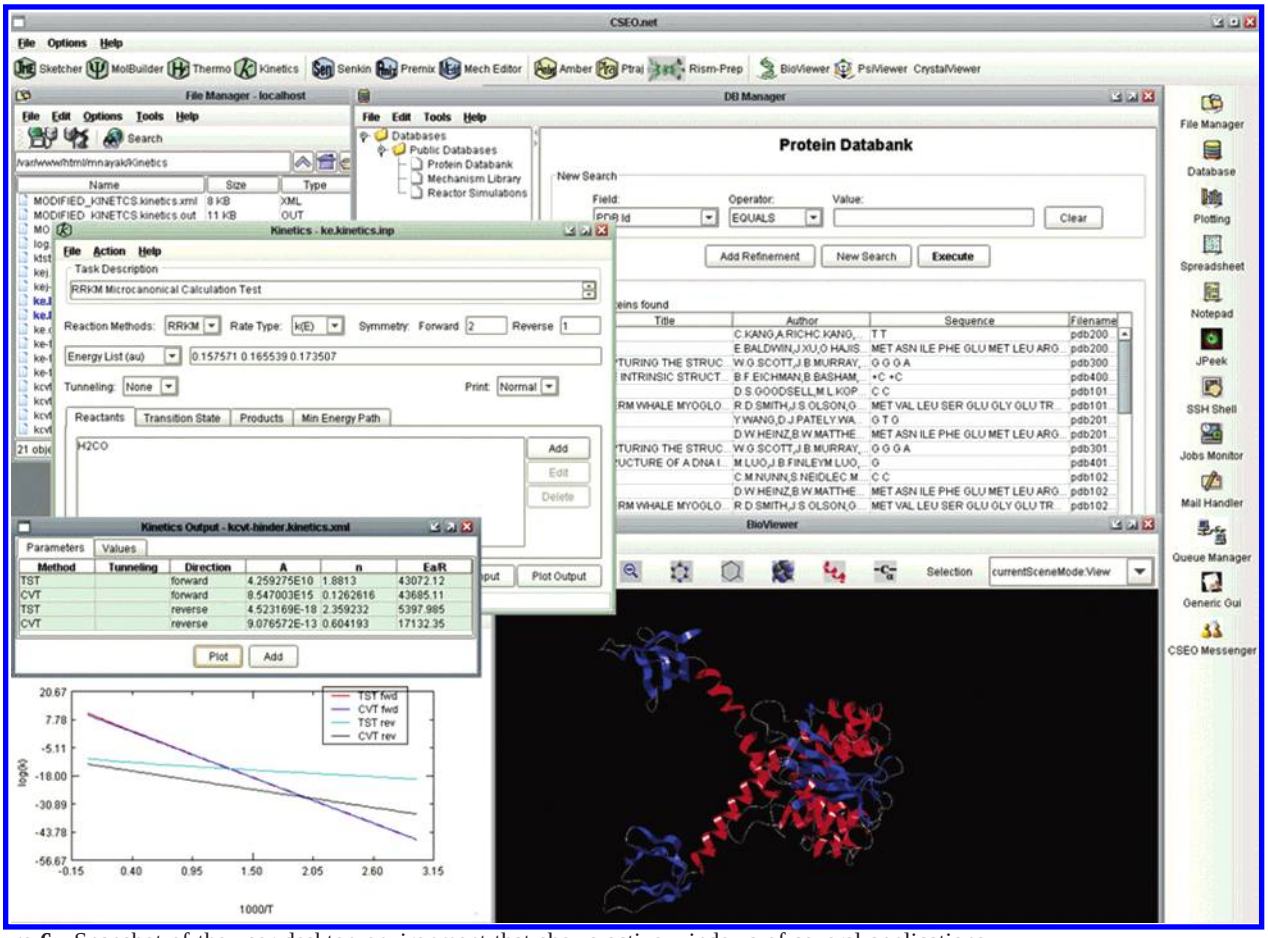

Figure 6. Snapshot of the user desktop environment that shows active windows of several applications.

into either the common or domain-specific tools. The common tools are tools that are useful to all domains, whereas the domain-specific tools consist only of tools that are specific to a particular domain. Figure 6 shows a snapshot of the user desktop environment that has active windows of several applications.

3. KNOWLEDGE MANAGEMENT SYSTEM

CSE-Online provides users with three different mechanisms for managing data. One is through the common file directory system via the File Manager. The second mechanism is via a multilevel database management system that allows users to store data in databases for query/retrieval at a later stage. Finally, CSE-Online has an electronic documenting system called eDoc for editing text and capturing data in different formats, and it stores them in different layers with links to the data viewers.

File Manager. The File Manager is a graphical user interface (GUI) that allows a user to manage files in their user directories on the local computer or the remote private server depending on the selected mode of operation. A snapshot of the File Manager is shown in Figure 7.

For the connection to a private server mode, when the environment is first activated, the user directory trees on the private server are created. The user can delete, copy, move, and create files on the server as well as upload files from the local computer to the server and vice versa. These tasks can be done with simple click-button or drag−drop mouse functions.

The File Manager is especially valuable for a remote user because communication to the server is made only when there is a request for adding, deleting, updating, or changing the content in the user directories and data storage areas. In addition, the File Manager can periodically check the content of the user directories and update the information when it is modified because of certain application processes running on the server (or even the remote desktop environment).

An advantage of the File Manager is that the communication with the server takes place when an actual file request is needed. If there is a network failure, the environment is not terminated because it runs on the local computer. However, any changes to the user directory on the server will not be made until the connection is re-established. The File Manager can cache the change and then resync those changes in light of any other changes that may have been made by others in the meantime. Thus, the environment is more stable with respect to network failure. The File Manager also allows the user to move files across different private servers in a transparent manner.

When no connection to any remote private server is made, as in the without connection to a private server mode, the File Manager maps file directories on the user's local storage devices.

Multilevel Database Management System. The multilevel database management system (MDMS) is complete with a front-end GUI, which provides end users with a user-friendly tool to manipulate scientific data in databases that physically exist on a remote server. The MDMS of CSE-Online is intended to be platform-independent, Web-acces-

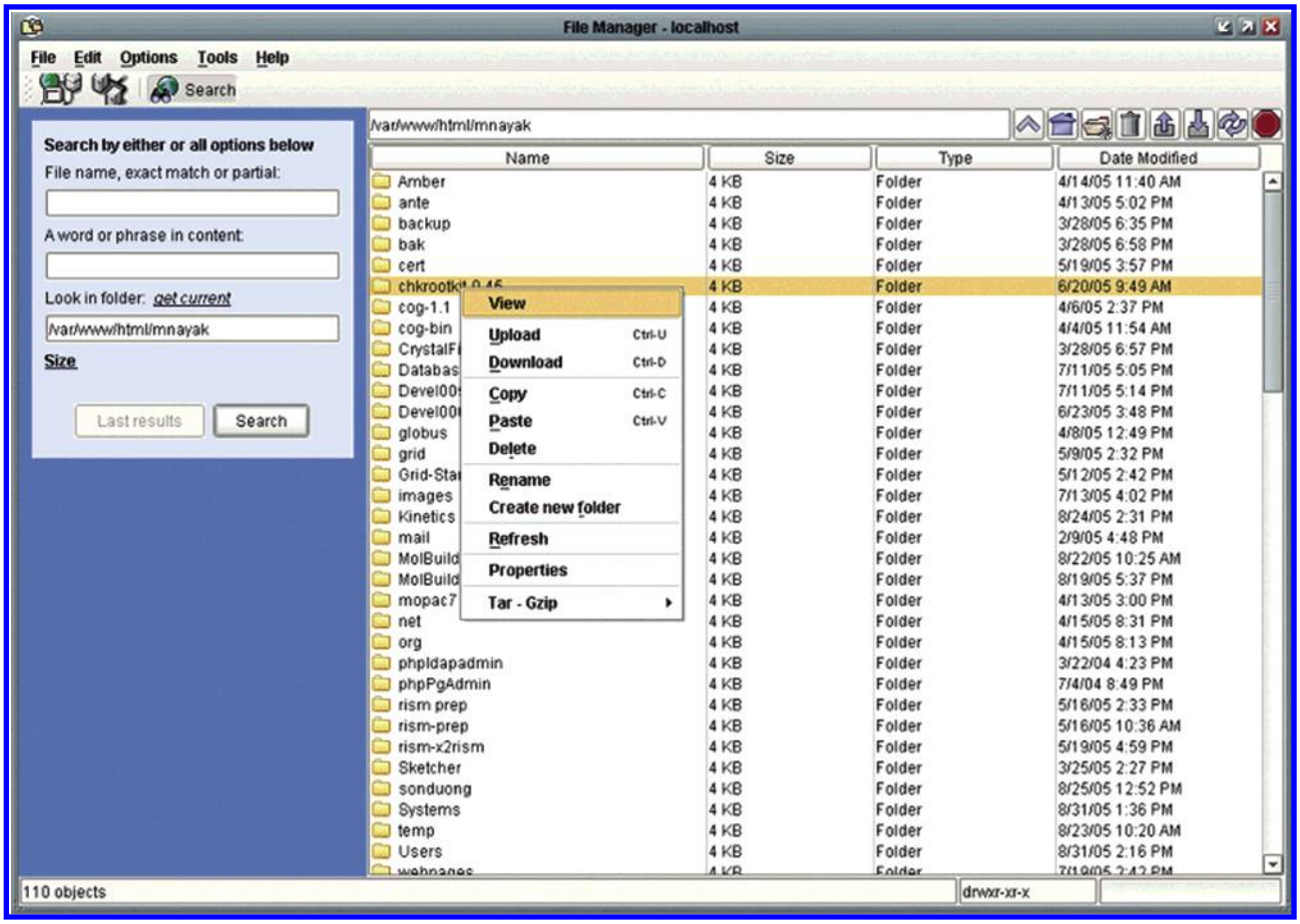

Figure 7. Snapshot of the File Manager.

sible, and scalable. It has the capability of allowing the user to customize their search queries. The data retrieved from the database as a result of the user-created queries is displayed in a manner that is useful to the end users, and the data can be accessible to different applications of CSE-Online; for example, Protein Databank (PDB) data can be viewed in the BioViewer tool. In fact, the retrieval data can interface with all of the applications in CSE-Online that can use or view such data.

MDMS has three levels of accessing data (open, limited group, and user-only access) corresponding to three kinds of databases (public, shared, and private databases, respectively). The public databases consist of data that has already been published and is available to the scientific community online. Every user of the CSE-Online environment can access these public databases and have the capability to query, retrieve, and analyze the data. These databases are maintained by CSE-Online. Users can submit information to be added to these public databases. A designated gatekeeper will validate such data before it can be posted.

Private databases contain proprietary data of individual users. The user can request the capability to create private databases using the available database schemas on the CSE-Online remote servers. It is possible to create private databases on the user's private server; however, installation of the CSE-Online database system on such a server is required. The private databases will allow the users to store and manage their own data.

Any user, who has the privilege to create private databases, will have the capability to create shared databases. The user can elect any of their private databases to be a shared database. The user is then able to designate specific registered CSE-Online users who can access such a database and assign privileges to these users. The shared databases reside on the CSE-Online server.

For query and retrieval, the user can select different search levels, namely, locally in the selected database or globally in the selected private databases and all accessible distributed public or shared databases which have the same structure. With a single query, the search is transparent to the user whether it is within a single private database or over a large number of distributed databases.

CSE-Online MDMS relies on the Java and Java Webstart technologies for the GUI. For the server-side components, it involves the standard Enterprise JavaBeans[29] with the JBoss Application Server, version 4.0.2.[32] The database system used is PostgreSQL[33] coupled with an object-relational management tool called Hibernate.[34]

Software Architecture and Implementation. Figure 8 shows the overall software architecture of the CSE-Online MDMS. The client has a "*Query, Retrieval, and Management component*", which connects over RMI to the remote servers in the system as shown. The remote private servers and public servers have the "*Data Access components*" to access data from the database and the file system, while the CSE-Online server has an additional component called the "*Server Lookup component*" to access information of the locations of all the remote servers in the CSE-Online network. The design of the database system consists of a three-tier architecture with the GUI tier residing on the client machine,

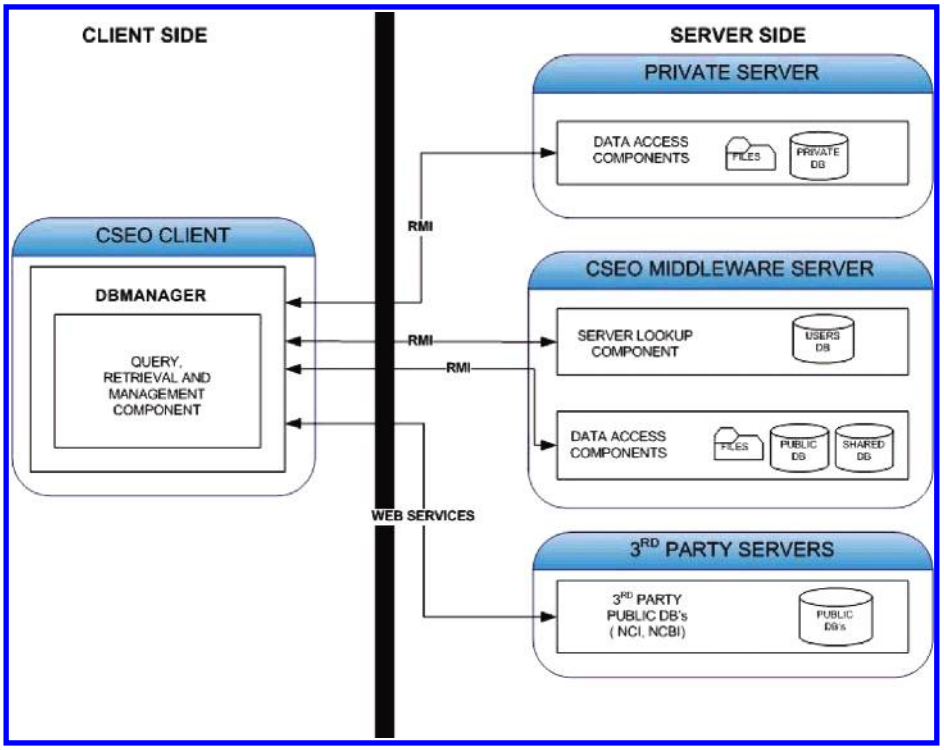

Figure 8. Plot of the software architecture of the Multilevel Database Management System (MDMS).

the EJBs and Java classes which form the Data Access and Server Lookup components forming the middle tier, and the database and the file system forming the third data tier in the application. In addition, the CSE-Online MDMS has a component that allows users to access external online public databases from within the CSE-Online environment. The current implementation is by using http; however, we plan to convert it to Web services at a later stage in development.

Current Functionalities. The CSE-Online MDMS currently has three public databases accessible to *every* user of the CSE-Online environment. These are the PDB database, the Reaction Mechanism database, and the Combustion Flame database. The PDB database is a mirror of the Protein Data Bank database from the Research Collaboratory for Structural Bioinformatics.[22] The Reaction Mechanism database is a collection of mechanisms for the combustion of hydrocarbons, including soot formation, existing in the literature. The Combustion Flame database is a collection of published experimental and theoretical data for different types of combustion flames. Currently, it has a large amount of premixed flame data. Data from these databases can be viewed and analyzed by tools available within the CSE-Online environment. For instance, data from the PDB database can be visualized using the BioViewer tool, as shown in Figure 9; data from the Reaction Mechanism database can be viewed and analyzed by the Mechanism Editor, and finally data from the Combustion Flame database is interfaced with the plotting tool, which displays plots of any selected data. We have also interfaced to several external online public databases for structural information, particularly, the National Cancer Institute (NCI) small molecule database[35] and the National Center for Biotechnology Information collection of public databases,[36] which include the Structure, 3D Domains, and PubChem databases. An illustration of the connection to the NCI database from within the CSE-Online environment is shown in Figure 10.

CSE-Online currently allows users to create two types of private databases, namely, the Chemical Properties and Reaction Mechanism databases. The Chemical Properties database allows users to manage molecular properties data obtained from quantum chemistry calculations. Users can insert, copy, paste, and delete records in their private databases. Users also have the option of copying and pasting records from one database to another, but only for databases sharing the same schema. Private databases are read/write as opposed to public databases, which are read-only. There are options available for deleting multiple records from the database as well as for deleting or renaming the entire database itself.

We have successfully prototyped the shared database component that allows users to share their private databases to designated users. Work is in progress to make this component fully functional.

We have designed a generic GUI that allows users to create their own specific queries for each database. The GUI allows the user to select the name of the field that they want to search and then apply an “equals” or “fuzzy” search on the search string that they enter. There is the possibility of adding additional terms to the search query by clicking on the “Add Refinements” button. Clicking on “Execute” will execute the query on the required database. For private databases, users have the option of customizing their queries by adding additional searchable fields into the database. These fields are called user-defined fields and allow the user to extend the implemented CSE-Online static searchable fields by inserting their own.

CSE-Online has the capability of searching across distributed databases, if all of the databases share the CSE-Online schemas by the means of the Server Lookup Component on the CSE-Online Server that contains the remote server locations of all accessible public databases. Upon activating the database tool, the connections are made

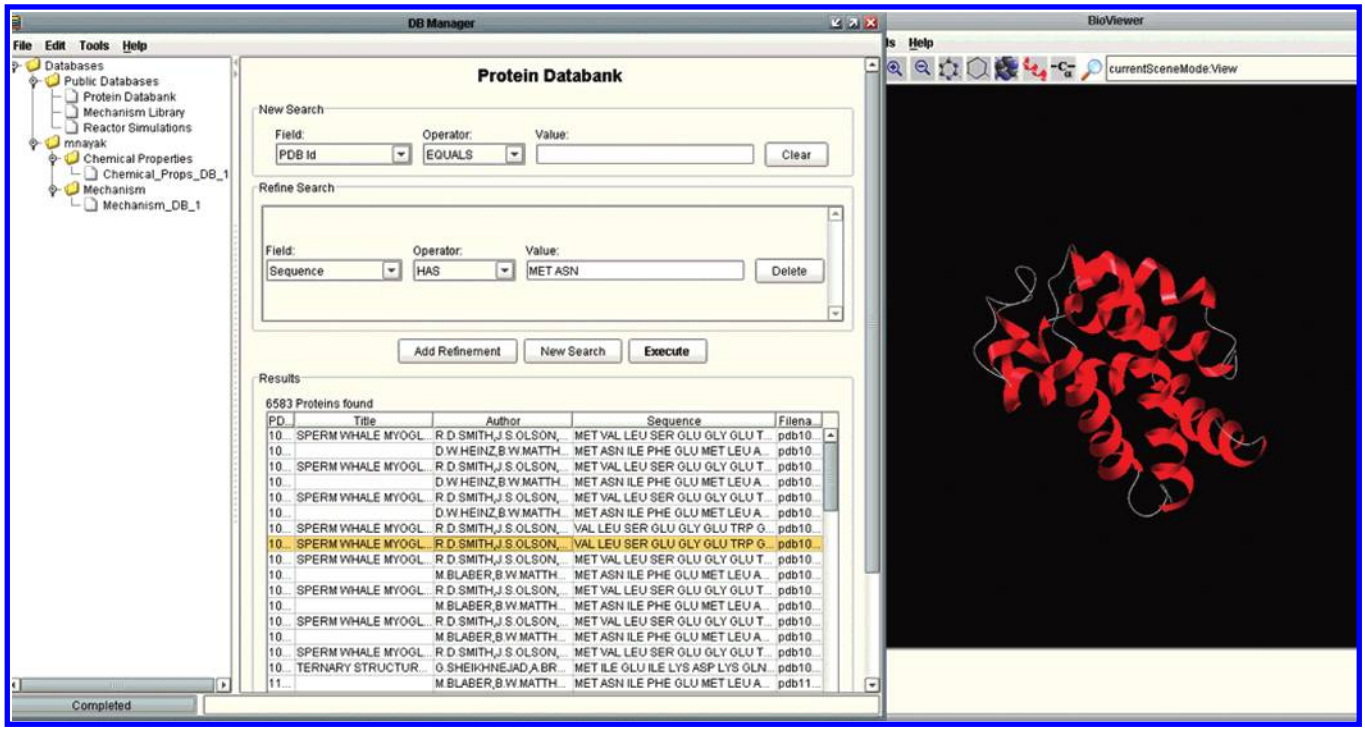

Figure 9. Snapshot of the MDMS connecting to the Protein Data Bank and having the retrieved data viewed by the BioViewer tool.

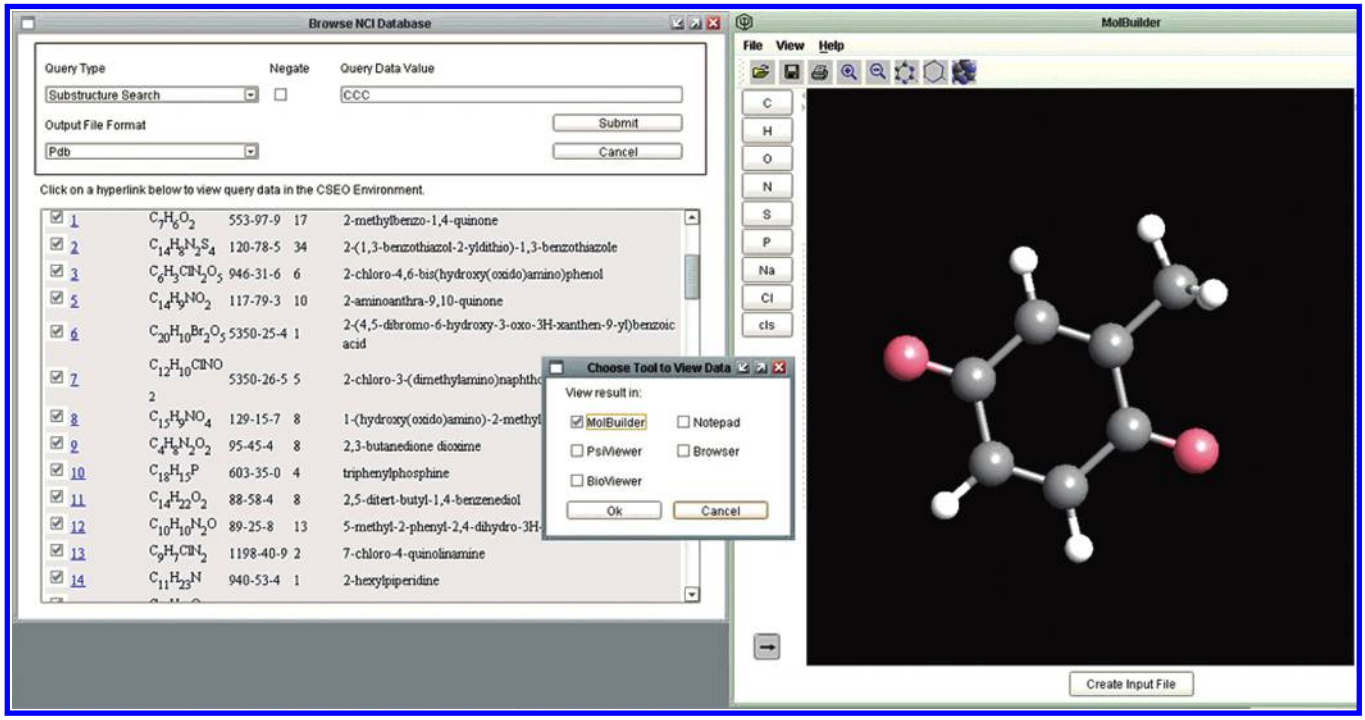

Figure 10. Snapshot to show the capability of MDMS to connect to the NCI public database. The selected data can be retrieved and viewed by any appropriate tools available within the CSE-Online environment.

to these different servers. The user can now execute a query, and the data is retrieved and displayed to them transparently. The user is not even aware that the GUI is accessing more than one database concurrently. This capability has recently been prototyped using multiple copies of our PDB database located on different servers.

Electronic Documenting System. The CSE-Online environment has a general-purpose online electronic documenting system called eDoc that is platform-independent and has the capability of editing text as well as capturing data in different formats (flat files, images, voice, etc.) and then storing them in different layers with links to data viewers and original data sources. eDoc is capable of recording the progress of a scientific research project with all of the associated data and their pedigree in a single multilayer electronic document.

The main feature of eDoc will be its multiple layers of data underneath a single document presented to the reader. The reader then has the ability to access all layers of data, if so desired. For example, an inserted figure can embed another layer of data, which contains the original data file and a link to the application tool that was used to generate

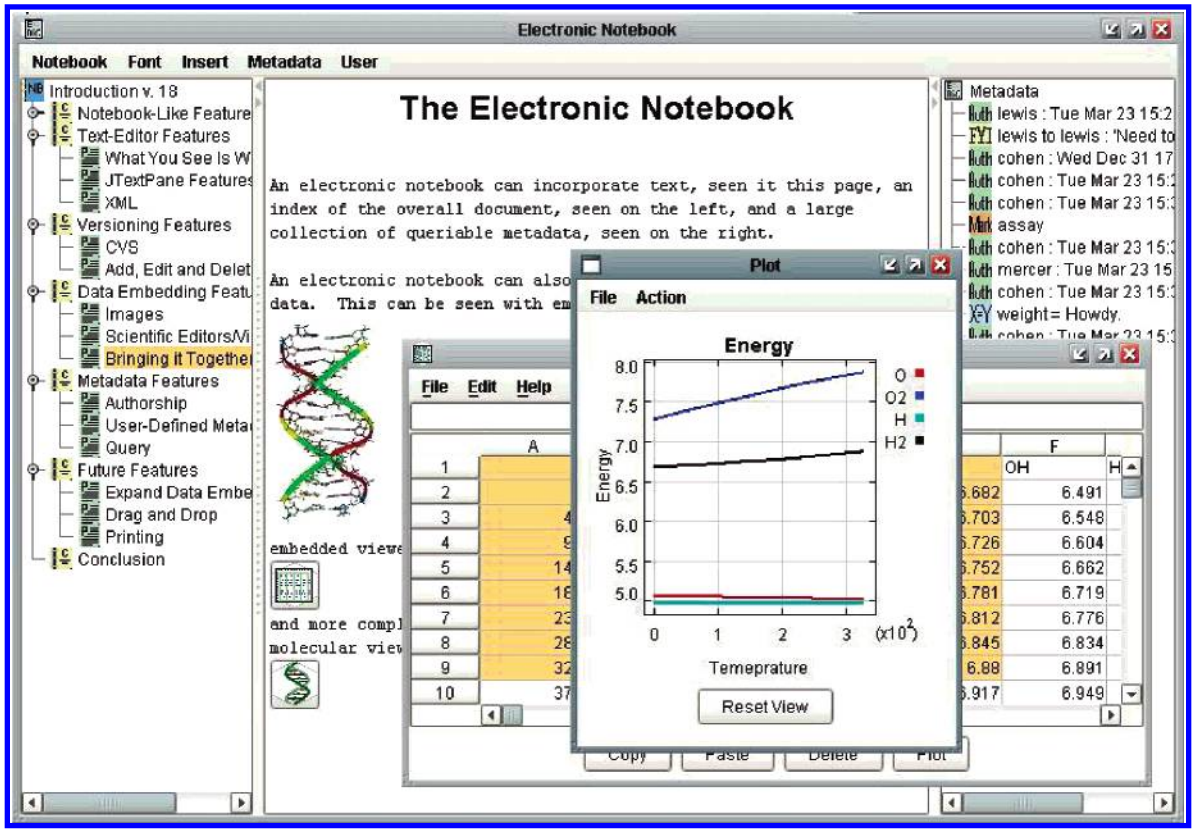

Figure 11. Snapshot of the prototype of the eDoc system displaying an electronic document with embedded data.

it. Existing technologies of what you see is what you get (WYSIWYG), confine users to seeing a two-dimensional view of the inserted figure. Such restrictions would hinder the exploration of different points of view and interpretations on a single object. Furthermore, most often, the inserted data are extracted from the original and many relevant data are lost during the documenting process. By allowing for more layers of data to be stored, eDoc preserves all relevant data during the documenting process.

In line with the CSE-Online overall software architecture, eDoc also has an open software architecture that allows different data viewers to be developed independently by third parties. Such viewers can also be loaded at run time.

Prototype. We have successfully prototyped the eDoc system. Users can edit text and insert figures. eDoc automatically links the figure with its viewer and original data files. This capability is illustrated in the snapshot shown in Figure 11. This snapshot shows an inserted plot opened by the plotting tool and the selected data used for plotting such a graph opened by the spreadsheet application. This illustrates the main feature of the eDoc system. It is still under active development, and more work is needed for supporting different scientific data types and viewers.

4. VISUALIZATION/ANALYSIS COMPONENT

Because the CSE-Online environment is Web-based, its visualization component must also be based on the Java platform-independent programming language. All 3D graphics applications in CSE-Online are based on the Java OpenGL (JOGL)[37] language and are deployed over the Internet onto the client using Java Webstart. JOGL is a cross-language interface or binding, providing an interface to native

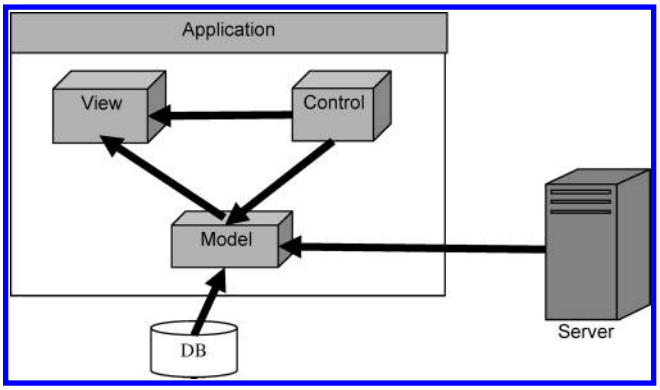

Figure 12. Plot of the software architecture of the 3D graphics application tools.

OpenGL and CG libraries using the Java native interface. JOGL is not part of the Java standard library. JOGL uses hardware acceleration and could achieve the performance of C++ OpenGL applications, and it provides all of the features supported by OpenGL. It is expected to become the de facto graphics library for the Java platform and to usher in the development of graphics applications and games in Java.

3D Graphics Application Software Architecture. 3D graphics applications in CSE-Online are developed around the Model−View−Control architecture using object-oriented design as shown in Figure 12. The Model component consists of application data—comprised of model data such as atoms, molecules, proteins, and visualization data. The View component consists of the 3D graphics objects, which handle the rendering of the graphics objects, and the Control component refers to the user interaction through the GUI and the mouse.

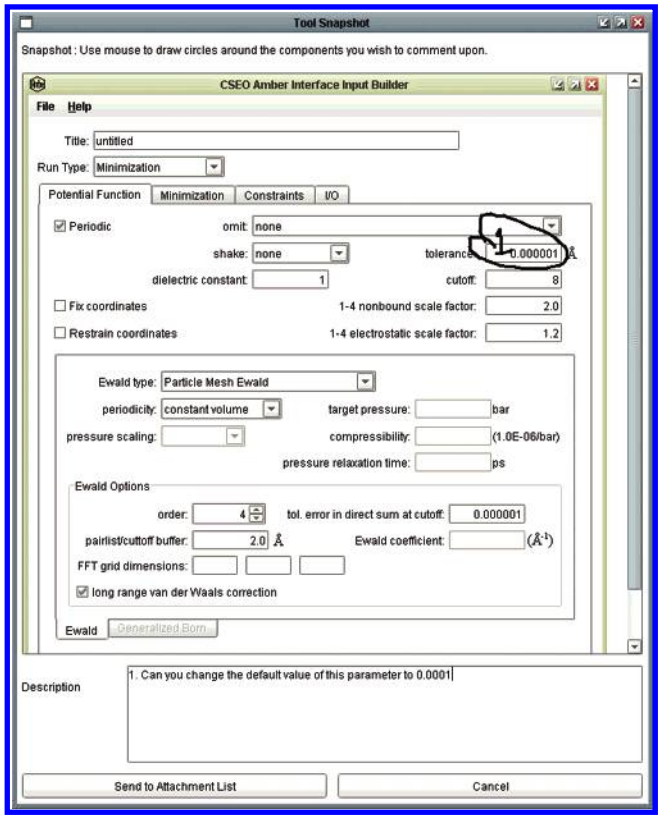

Figure 13. Snapshot of the User Feedback System.

The object-oriented design of the software allows sharing the Model and View component classes among all 3D graphics applications in CSE-Online, despite JOGL not being inherently object-oriented. The commonality of the classes also results in interoperability among various graphics applications, apart from decreasing the development effort. The graphics APIs consist of common classes for the Graphics-Canvas/Renderer and Graphic objects. Individual applications consist of classes derived from the common classes. The common base classes ensure compatibility among various applications and enforce standards. The Graphic object consists of functions for defining the graphic scene and picking process; a new graphics class is derived from the generic Graphic object, and its scene, picking, and other functions are over-ridden. These derived graphics objects are delegated to the graphics canvas, which manages the scenes. The Graphic objects use display lists (OpenGL feature) for optimizing the OpenGL rendering of scenes. The mouse handling functions in the Graphics-Canvas object is over-ridden to implement any application-specific mouse interaction. Currently, we have a number of 3D graphics applications, such as MolBuilder for building a 3D molecule from atoms and fragments, PsiViewer for visualization results from quantum chemistry calculations, and BioViewer for the visualization of biomolecules.

5. COMMUNICATION COMPONENT

CSE-Online provides capabilities for both synchronous and asynchronous modes of communication. For asynchronous communication, an e-mail handler tool based on the Pooka tool [38] that has POP and IMAP support for incoming mail is used. It can also handle sending mail using the SMTP protocol. In addition, the communication component also has a user feedback system that allows a user to communicate with the developers or other users on issues associated with a particular application tool. For synchronous communication, we have developed a CSEO Messenger that allows users to have both text and audio chat capabilities with other users.

User Feedback System. The user feedback system allows users to take a snapshot of any particular tool in the CSE-Online environment, make a note on the picture, save all associated data needed to recreate the same state of the tool, and write comments to send to others by e-mail or to send to the development team. This provides an effective mechanism for users to communicate directly with the development team to give feedback, bug reports, and suggestions for further improvements. Figure 13 shows a snapshot of the Sander GUI with a note made by the user to request changing a parameter on the GUI. It is interesting to note that, for most if not all scientific software developments, no such automated feedback mechanism exists.

CSEO Messenger. CSEO Messenger will enable users to communicate with each other via text, audio, or video regardless of their geometric positions. Currently, it provides instant text and audio exchange among users on the Internet. CSE-Online communication's main design principles include platform independence, real-time transmission, and system tolerance. The software architecture of CSEO Messenger is shown in Figure 14. The system is built on the well-known Java media framework (JMF). JMF provides a unified architecture and messaging protocol for managing the acquisition, processing, and delivery of time-based media data.[39] By exploiting the advantages of the Java platform, JMF provides a common cross-platform Java API for accessing underlying media frameworks. Moreover, JMF satisfies our real-time demand because it is built on real-time protocol (RTP). Figure 14 shows the software architecture of the system. We provide both the peer-to-peer (P2P) and via a relay server modes of communication. This is necessary because, while the P2P mode can reduce the load of the server and the network by direct connection between the peers and is used as the primary operation mode, P2P connections between the peers over the Internet are not always possible. In our system, we used the "hole punching" technique[40] to establish the P2P user datagram protocol (UDP) connection between two peers. Currently, about 82% of network address translators (NATs) support such a hole punching for UDP and 64% do so for a transmission control protocol (TCP) connection. The remaining NATs that do not support such a hole punching require the use of a relay sever as an alternative mode of operation. In this relay server mode, the TCP protocol is used to transfer text and control data to ensure the reliability, whereas the RTP/UDP protocol is used for audio/video data. The use of a relay server is also essential for the future expansion of CSEO Messenger to allow for cyberconferencing to facilitate collaborations among users.

The future work of CSE-Online communication will focus on building a virtual classroom, shared windows, and remote control so that users can exchange not only text and audio but also images, video, and especially remote manipulation of the data or even the application tool.

6. RESOURCE MANAGEMENT SYSTEM

CSE-Online currently supports three modes of running an application: (1) on the log-in private server, (2) on another

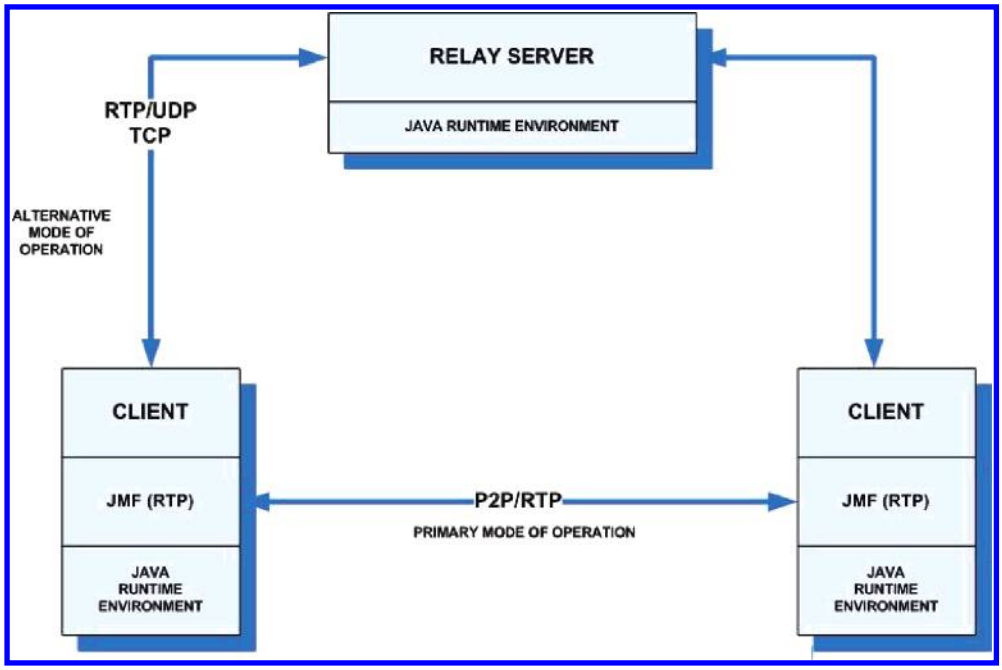

Figure 14. Plot of the software architecture of the CSEO Messenger for text and audio/video communication between two users.

remote Unix/Linux computer or cluster, and (3) on a Globus-enabled computing grid such as the TeraGrid. Our current resource management system consists of the queue manager, which is the front end to the computing environment accessible to the user from within the CSE-Online environment. This GUI allows users to work with any PBS-enabled server using a user-friendly and intuitive interface. The tool allows for submission, deletion, listing of jobs, and other PEB functionalities. Users can transfer files to and from the PBS server to their local machines or the server they were initially logged on to through the CSE-Online login. A complete list of PBS functionalities provided by the queue manager is as follows: (1) submission of jobs; (2) deletion of jobs; (3) viewing details of a job; (4) order, move, hold, or release a job; and (5) view output files.

The queue manager tries to automatically detect whether the server that you are connected to is PBS-enabled. If it cannot find such a service, the user can use the preferences option to set the location of the PBS commands. The tool also allows for the submission of jobs to any queue on the PBS server if it has more than one queue. Currently it does not have a metascheduling capability. Our future plan is to interface the queue manager with an available metascheduler to make effective use of distributed computational resources.

The queue manager has also been expanded so that users can submit jobs to the TeraGrid. Users can log in to any of the interactive TeraGrid nodes and submit jobs. The queue manager uses the cog toolkit[41] from Globus technologies to submit jobs to any Globus-enabled grid machine. This toolkit makes use of the Java Web-services architecture to submit jobs to grid (Globus-enabled) machines from the local computer. Because this toolkit was written with Java interfaces, it can be incorporated into the CSE-Online desktop environment.

7. RELATIONSHIP WITH OTHER DEVELOPMENTS

As mentioned earlier, there are a number of developments within the modeling and simulation domain that have similar aims, such as CMCS,[11] GEMSTONE,[12] and GridChem.[13] All of these Web-based simulation environments share the common Web-services software architecture and, thus, have its advantages and limitations, as mentioned in the Introduction. What distinguishes CSE-Online from these other developments is its unique capability to concurrently access multiple *private* servers located at different locations. It is important to point out that it is possible to leverage the current CSE-Online functionalities with those from the above developments and other Web-services-based tools such as P-Grade for access to Globus resources,[42] OGSA-DAI for distributed databases,[43] AccessGrid for video conferencing,[44] Jabber for instant messaging,[45] and so forth by interfacing CSE-Online with these tools or environments using its open architecture and APIs. This would greatly enhance the CSE-Online environment and avoid duplicating the efforts.

8. CONCLUSION AND PERSPECTIVE OUTLOOK

CSE-Online provides a unique and ubiquitous desktop environment for research, education, and collaboration in computational science and engineering. It can be accessed from a Web browser regardless of geophysical location or time zone and does not require any installation beyond the Java Runtime Environment on the user's local computer. It is an extendable, platform-independent, and secured environment that enables a seamless interface and data flow between a variety of different computational application tools and databases. It provides a complete bottom-up environment that integrates methods from different disciplines while having the look and feel of a single desktop application. It has different modes of operations such as the connection to a private server, without connection to a private server, and offline modes to support different working environments and styles. It allows the user to access remote data and computing resources in a transparent manner. Its knowledge management system provides different mechanisms for the user to manage scientific data in the form of files or databases as well as share certain data with other designated users. The

environment also provides different communication tools, in both synchronous and asynchronous modes, to facilitate training, user support, and collaboration. Its 3D graphics tools can provide real-time performance for the visualization and analysis of data from remote sources.

The CSE-Online open software architecture allows third-party development of different desktop application tools. Furthermore, the desktop applications are loaded at run time, thereby allowing the user to select only applications that are needed to build the executable. From the software engineering point of view, this architecture allows the development to be scalable and the desktop environment to support an unlimited number of applications while the memory requirement of the environment depends only on those selected by the user.

CSE-Online, at the beginning, was only an online desktop environment for computational science and engineering; however, it is evolving into middleware that can provide a framework for accessing and managing remote data and resources including the computing grid for any domain, not necessarily just within computational science and engineering.

ACKNOWLEDGMENT

This project is supported by the National Science Foundation under the Information Technology Research Initiative. We are grateful to our Advisory Board members, Professors Jack Simons, Monte Pettitt, Adel Sarofim, Gary Lindstrom, Martin Berzins, and Chris Johnson, and Drs. Jeff Nichols and Max Tirtowidjojo for their guidance.

REFERENCES AND NOTES

(1) 2003 NSF Blue Ribbon Advisory Panel Report on Cyberinfrastructrure. www.cise.nsf.gov/evnt/reports/toc.htm (accessed Sept 2005).
(2) TeraGrid. http://www.teragrid.org/ (accessed Sept 2005).
(3) Barrett, D. J.; Silverman, R. E.; Byrnes, R. G. *SSH: The Secure Shell—The Definitive Guide*, 2nd ed.; O'Reilly: Sebastopol, CA, 2005.
(4) X11 Technologies. http://www.x.org/X11.html (accessed Sept 2005).
(5) Web Services Architecture Requirements. http://www.w3.org/TR/wsa-reqs/ (accessed Sept 2005).
(6) Bruzzone, A. G.; Uhrmacher, A.; Page, E. H. *Proceedings of the 1999 International Conference on Web-Based Modeling and Simulation*; The Society for Computer Simulation International: San Diego, CA, 1999.
(7) Charnes, J. M.; Morrice, D. J.; Brunner, D. T.; Swain, J. J. *Proceedings of the 1996 Winter Simulation Conference*; ACM Press: New York, 1996.
(8) Fishwick, P. A.; Hill, D. R. C.; Smith, R. *Proceedings of the 1988 International Conference on Web-Based Modeling and Simulation*; The Society for Computer Simulation International: San Diego, CA, 1998.
(9) Page, E. H.; Buss, A.; Fishwick, P. A.; Healy, K. J.; Nance, R. E.; Paul, R. J. Web-Based Simulation: Revolution or Evolution? *ACM Trans. Model. Comput. Simul.* **2000**, *10*, 3−17.
(10) Kuljis, J.; Paul, R. J. An appraisal of Web-Based Simulation: Whither We Wander. *Simul. Practice Theory* **2001**, *9*, 37−54.
(11) Collaboratory for Multi-Scale Chemical Science. http://www.cmcs.org (accessed Sept 2005).
(12) GEMSTONE. http://griddevel.sdsc.edu/gridsphere/gridsphere?cid=gemstone&JavaScript=enabled#tab1 (accessed Sept 2005).
(13) Computational Chemistry Grid, GridChem. https://www.gridchem.org/ (accessed Sept 2005).
(14) Discovery Net. http://ex.doc.ic.ac.uk/new/index.php (accessed Sept 2005).
(15) Combechem. http://www.combechem.org/ (accessed Sept 2005).
(16) Reality Grid. http://www.realitygrid.org/ (accessed Sept 2005).
(17) Virtual Laboratory Project. http://www.gridbus.org/vlab/ (accessed Sept 2005).
(18) North Carolina Board of Science Technology and National Research Council. *Collaboratories: Improving Research Capabilities in Chemical and Biomedical Sciences: Proceedings of a Multisite Electronic Workshop*; National Academy Press: Washington, D.C., 1999.
(19) Agarwal, D. A.; Sachs, S. R.; Johnston, W. E. The Realities of Collaboratories. *Comput. Phys. Commun.* **1998**, *110*, 134.
(20) Kouzes, R. T.; Myers, J. D.; Wulf, W. A. Collaboratories: Doing Science on the Internet. *IEEE Computer* **1996**, *29*, 8.
(21) Kiernan, V. Internet-Based 'Collaboratories' Help Scientists Work Together. *The Chronicle of Higher Education*, March 12, 1999.
(22) Research Collaboratory for Structural Bioinformatics. http://www.rcsb.org (accessed Sept 2005).
(23) Space Physics and Aeronomy Research Collaboratory. http://www-.si.umich.edu/sparc/ (accessed Sept 2005).
(24) Materials Microcharacterization Collaboratory. http://tpm.amc.anl.gov/MMC/ (accessed Sept 2005).
(25) Environmental Molecular Science Collaboratory. http://collaboratory.emsl.pnl.gov/ (accessed Sept 2005).
(26) Truong, T. N. In *An Integrated Web-Based Grid-Computing Environment for Research and Education in Computational Science and Engineering*, 37th Annual Simulation Symposium, Arlington, VA, April 18−22, 2004; p 143.
(27) Truong, T. N.; Cook, T.; Nayak, M.; Boonyasiriwat, C.; Tran, L.-T. T.; Zhang, S. Computational Science and Engineering Online: An Integrated Web-based Environment for Multi-Scale Modeling of Complex Reaction Systems. *Mol. Phys.* **2004**, *102*, 353.
(28) Computational Science and Engineering Online (CSE-Online). http://cse-online.net (accessed Sept 2005).
(29) EJB 3.0. http://www.jcp.org/en/jsr (accessed Sept 2005).
(30) Hypertext Transfer Protocol—HTTP/1.1, RFC 2068.
(31) Marinilli, M. *Java Deployment Using JNLP and WebStart*; Sams: Indianapolis, Indiana, 2001.
(32) JBoss Application Server. http://www.jboss.org/products/jbossas (accessed Sept 2005).
(33) Postgres SQL. http://www.pervasive-postgres.com (accessed Sept 2005).
(34) Hibernate. http://www.hibernate.org (accessed Sept 2005).
(35) National Cancer Institute Small Molecule Database. http://dtp.nci.nih.gov/docs/3d_database/Structural_information/structural_data.html (accessed Sept 2005).
(36) National Center for Biotechnology Information. http://www.ncbi.nlm.nih.gov/ (accessed Sept 2005).
(37) Java OpenGL (JOGL). http://jogl.dev.java.net (accessed Sept 2005).
(38) Pooka E-mail Client. http://sourceforge.net/projects/pooka/ (accessed Sept 2005).
(39) *Java Media Framework API Guide*; Sun Microsystems, Inc.: Santa Clara, CA, 1999.
(40) Ford, B.; Srisuresh, P.; Kegel, D. In *Peer-to-Peer Communication Across Network Address Translators*, The 2005 USENIX Annual Technical Conference (USENIX '05), Anaheim, California, 2005; pp 179−192.
(41) Open Grid Service Architecture. http://www.globus.org/ogsa (accessed Sept 2005).
(42) Kacsuk, P.; Kiss, T.; Goyeneche, A.; Delaitre, T.; Farkas, Z.; Boczko, T. A High-Level Grid Application Environment to Grid-Enable Legacy Code. *ERCIM News*, November, 2004.
(43) OGSA-DAI. http://www.ogsadai.org.uk (accessed Sept 2005).
(44) AccessGrid Videoconferencing. http://www.accessgrid.org (accessed Sept 2005).
(45) IETF, Extensible Messaging and Presence Protocol, IETF RFC 3920. http://www.faqs.org/rfcs/rfc3920.html (accessed Sept 2005).

CI0503917