CONCURRENT ENGINEERING: Research and Applications

Internet-based Distributed Collaborative Engineering Analysis

Qiuli Sun* and Kurt Gramoll†

School of Aerospace and Mechanical Engineering, 865 ASP Avenue, Felgar Hall, Room 212, The University of Oklahoma Norman, Ok 73019-0601, USA

Abstract: This paper proposes an engineering analysis environment that allows remote users to conduct three-dimensional finite element analysis collaboratively through the Internet. Java and Java 3D were chosen to develop the working prototype due to their advantages of platform-independence and network supporting. The environment allows remote users to work collaboratively on the same analysis object simultaneously. It reads the geometric data generated by the collaborative geometric modeling environment. The user can interact directly with the geometric model to perform operations, such as applying, editing, and deleting boundary conditions and forces. The operations are propagated among the team members, which creates a distributed shared environment. The commands are transmitted instead of the generated data, and thus the network traffic associated with the collaboration is minimized. Different from classical server/client models, the environment adopts a strategy in which the client-side application has full analysis capabilities while the server only manages communication. The essential features for distributed collaboration are discussed. The actual design consideration of the working prototype is presented to help illustrate the complexity and development of the collaborative environment. The environment is open to the public at www.vcity.ou.edu.

Key Words: internet-based, collaborative, engineering analysis, distributed, finite element, remote.

1. Introduction

The expanding use of the Internet has provided tremendous possibilities for engineering design and analysis. The omnipresence of the Internet has made distributed collaborative engineering design and analysis possible, which means that geographically dispersed engineers can complete design and analysis tasks jointly through the Internet [1]. This paper proposes an engineering analysis framework that allows remote users to conduct finite element analysis, including pre- and postprocessing, collaboratively over the Internet. Java and Java 3D were chosen to develop the engineering analysis framework since they provide excellent performance, support networking and 3D, and can be safely downloaded to run on client machines. In a second paper, the distributed collaborative geometric modeling was discussed [2].

The needs of industry, as well as engineering education, were major reasons to develop the collaborative engineering analysis environment. As engineering design becomes increasingly complex, an individual engineer or a single company may not able to complete an entire design task alone. Concurrent engineering and collaborative engineering are therefore needed to coordinate product development that involves designers from different departments in the same company, as well as from different companies [3,4]. The designers and experts are, however, often geographically separate, and it is almost impossible to move them all to the same working location. Traditionally, meetings, faxes, and phones are used to coordinate design work. These methods can cause misunderstandings, delay of information transfer, and design conflicts because complete product design information is not accessible to designers and experts when needed. It is therefore essential to design an environment that can support the key concepts of both concurrent and collaborative engineering. Industry has actually realized the importance of online smooth collaboration. For example, Bill Gates of Microsoft said in an email sent to developers and information technology professionals that he envisioned an online world where constellations of Internet-based services could collaborate seamlessly [5].

Other researchers have investigated distributed collaboration for engineering design, and a large number of papers on this field have been published. Senin, Pahng, and Wallace et al. proposed a distributed object-based modeling and evaluation (DOME) framework for product design [6]. Case and Lu proposed a discourse model used in software environments that provides automation support for collaborative

*Author to whom correspondence should be addressed.
†E-mail: gramoll@ou.edu

engineering design [7]. The discourse model treats interactions between designers as a process of discourse. DOME and the discourse model did not support real-time collaboration. Lee et al. proposed a prototype to implement web-enabled feature-based modeling in a distributed environment [8]. Cybercut is one of the first web-based design systems for fabrication [9]. Mori and Cutkosky proposed an agent-based prototype implementation in the design of a portable CD player [10]. Likewise, synchronous collaboration was not implemented in Lee's prototype, Cybercut, and Mori's agent-based prototype. To share CAD information seamlessly across an enterprise, major CAD suppliers have introduced a software system called Product Development Management (PDM), such as Windchill from PTC [11] and OneSpace from CoCreate [12]. This software system extends CAD data not only to nondesign departments of companies such as analysis, tooling development, manufacturing, testing, quality control, sales and marketing, but also to suppliers and partners of these companies. This software system currently, however, centers on asynchronous collaboration.

Although work has been done in this field, existing environments are still in their infancy. Few have implemented real-time collaboration. The product design is the focus of the current work, and research on distributed collaborative engineering analysis has been minor. This paper presents a collaborative environment for 3D finite element analysis over the Internet.

It would be ideal if mature geometric modeling technologies could be used in this research. However, at the time this research was conduced, commercially available CAD/CAM and PDM programs would not allow real-time collaboration and editing of documents over the Internet. Furthermore, commercial programs do not allow third parties to access their coding to integrate such tools with their programs. This research was to demonstrate the concept of collaborative engineering analysis over the Internet and not to integrate it into a specific commercial program.

2. Essential Features for Distributed Collaboration

To develop a collaborative environment for engineering design and analysis, it is important to identify its fundamental features derived from the expectations of the end users. Different researchers may have different viewpoints, but this paper focuses on the discussion of essential features, such as sharing information, natural communication, manipulation of design objects, database management, generation of design documents, intelligence, guaranteed real-time delivery, security, and scalability.

2.1 Sharing Information

It is vital for authorized users to share the latest information and access previous information in a collaborative environment [13]. Design information not only includes text descriptions of current design status but also engineering information such as 3D models, engineering drawings, and analysis results. In engineering design, the sharing of 3D models is particularly important because it is the best way to present the design objects. Since the design objects are continuously being updated while the design is going on, the environment should ensure that the users always obtain the latest design information. If the users want to review a previous version, they should also be able to access it seamlessly. Also, to increase the responsiveness of the collaborative environment, it is critical to share commands instead of the full generated data. It is not efficient to transmit large volumes of generated data over the Internet to update each remote computer, but a command is small, which can be used to generate the data on the local computer using the local CPU.

2.2 Natural Communication

Since design is essentially a collective work that requires many individuals to work together, it is important that a distributed environment is capable of helping users work collaboratively. Collaboration is the core concept of concurrent engineering and the heart of team-based design [3,4]. To complete a design, negotiations among participants need to take place continuously. Because designers are geographically dispersed, the environment needs to create a naturally collaborative environment in which each authorized participant can communicate with each other without obstacles [14]. For synchronous collaboration, it is ideal to provide a face-to-face virtual world where the participants can interact as if they were in the same room. For example, the participants in the virtual world can point to the same design object with verbal and nonverbal communication. Since different regions are in different time zones, it is also important to allow the user to communicate asynchronously.

2.3 Manipulation of Design Objects

To collaborate efficiently in a synchronous way, the collaborative design environment should not only create a virtual world for natural communication but also provide the capability for participants to manipulate the same design object simultaneously [14]. Whether the design object is a 3D model or a 2D drawing, when one of the participants is manipulating the design object other dispersed participants should see the action of the manipulation. If someone else makes a change to the design object, other should see it immediately on their own computers. This is an essential feature for synchronous collaboration.

2.4 Database Management

A database-management system (DBMS) should also be used to record design information and keep the latest and previous design and specifications [15]. There are numerous reasons for using a DBMS. First, a DBMS does not have concurrent-access anomalies. This is important because many users may access the data at the same time. Second, it can remove data redundancy and inconsistency. In a distributed multiuser environment, it is unwise to keep multiple working copies of the data because what starts as the same data may disagree over time. Third, a DBMS can reduce the difficulty in accessing data. Since the database keeps all of the latest information related to the design, the user can obtain any information from the database efficiently and conveniently. Also, the user can search the database for specific information, which includes documents regarding critical design decisions, specifications, and analysis results. Fourth, using a DBMS, it is easy to enforce security constraints on different users compared to a conventional file-processing system. Finally, it can ensure data transaction to be atomic, which means the data transfer must happen in its entirety or not all.

2.5 Generation of Design Documents

As many participants may work on the same design object, the environment should automatically record who makes the change when changes occur [16]. When a critical design decision is made, the environment should remind the participants to provide the reasons supporting the design decision. The design document should be stored along with other design document and 3D models as a part of the design object. This is extremely helpful when a design object needs to be redesigned or a similar design needs to be completed in the future. In this case, future designers can search information of similar designs. There is no need for them to spend time in tracking related documents. Carefully generated design documents will encourage reusing the design knowledge and increase the productivity of the designers.

2.6 Intelligence

A distributed design environment should be an intelligent design support system [14]. Due to the distribution of the participants, the environment should be able to process routine work automatically, such as notifying all of the interested participants when a critical change happens to a design object. The environment should guarantee that this notification is read by all of the participants in time. If someone disagrees with the change, a meeting should be scheduled through the use of the collaborative environment. The core argument should be recorded along with the design object for future review. Because design knowledge is routinely captured, the environment should be able to provide knowledge support to the designer when similar designs are available. If knowledge support of the environment is efficient, the development cycle of products can be shortened.

2.7 Guaranteed Real-time Delivery

A distributed collaborative environment is a real-time application, which is different from traditional data applications, such as Telnet, FTP, and Web browsing. This requires the quality of service from the Internet to be high so that lag time is minimized [18]. The packets sent should reach their destinations in a specified time interval, for instance, 100 ms. This also requires the Internet to promise guaranteed packet delivery since it is unacceptable for packets sent by one user not to reach their destinations. Currently, the Internet only implements a best-effort service model, which tries its best to deliver the packets but without any guarantee. The Internet with necessary quality of service may be available in the near future [17].

2.8 Security and Scalability

In a distributed design environment, privileges should be limited to needs of the users [18]. Some should be only allowed to view the general information such as the outer shape, while others should be allowed to see the detailed design. Data transferred over the Internet should be protected carefully to prevent sensitive information from being compromised. Security strategy is critical to the success of distributed design environments because no company will adopt a distributed environment that does not protect sensitive information sufficiently. As the number of the participants is constantly changing, a distributed design environment should allow scaling while maintaining an acceptable level of performance [15].

Generation of design documents, intelligence, guaranteed real-time delivery, security, and scalability are not the research focus of this paper. This is because generation of design documents and intelligence are usually the focus of design support systems while security issues and scalability as well as quality of service are in the realm of computer science. This paper centers on the implementation of sharing information and manipulation of design objects as they impact engineering design and analysis. Communication is achieved by providing real-time text-based discussion. Natural communication is not considered because it requires expensive hardware and a high Internet bandwidth. A database is used to manage the geometric files.

3. Overview of the Collaborative Environment

Three essential features of distributed collaboration: sharing information, manipulation of design objects, and communication are emphasized in the collaborative engineering analysis environment presented here. The environment implements real-time information sharing and real-time manipulation of the design objects among remote users. These features are different from other systems, such as DOME, and PDM systems which usually centers on asynchronous collaboration and offers capabilities such as publishing product information, viewing product information and collaboration notifications.

The environment contains two components: client-side applications and server-side applications (Figure 1). The first is a client-side Java applet, which is used to conduct finite element analysis. To make the interface easy to use, buttons in the applet are grouped into three toolbars according to their functions: top toolbar, left toolbar, and bottom toolbar (Figure 2). These three toolbars provide button-driven actions. In addition, the user can also employ the mouse to rotate, translate, and zoom in on or out from the object. For example, if the user clicks and drags the mouse, the user can rotate the object. This is useful because these actions are not associated with any button. The second component is a Java-based multiuser server and a database server, which manage communication and are the same servers utilized in the geometric modeling environment [2]. Employing the same servers can reduce the number of programs and hence minimize programming tasks.

The engineering analysis environment is a CAE program that can import the model generated by the geometric modeling environment, conduct typical finite element analysis, and visually demonstrate the computational results. The collaborative geometric modeling environment is a simple CAD application that provides a general-purpose 2D drawing tool, 3D operations, and a rendering tool (Figure 3). The main difference from traditional CAD programs is that, its internal design supports multiuser capabilities. Remote users within the same team can work on the same geometric model when they are geographically dispersed. For example, they can build, change, delete, and rotate the same geometric model. The model is saved as a file on the server and managed by a database. B-Rep data structures are used to describe the boundary of the model: the vertices, the edges, and the faces. Detailed discussions about the collaborative geometric modeling can be found in a recently published paper [2].

The difference between this collaborative analysis environment and traditional CAE programs is that the environment is Internet-based and supports multiuser

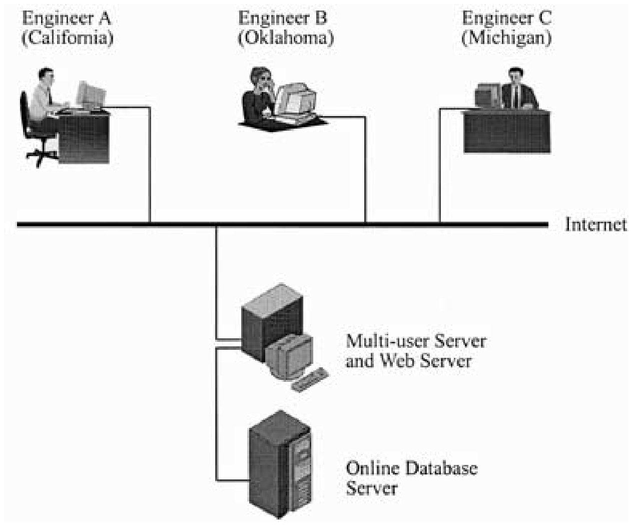
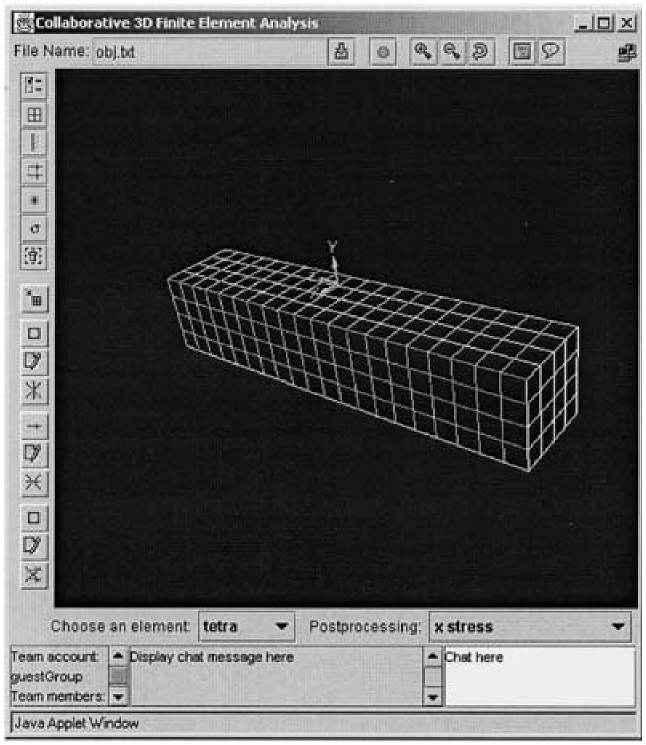


Figure 1. Architecture of distributed collaborative analysis environment.


Collaborative 3D Finite Element Analysis

File Name: obj.txt

Choose an element: tetra Postprocessing: x stress

Team account: guestGroup

Team members:

Display chat message here

Chat here

Java Applet Window

Figure 2. Interface of collaborative engineering analysis environment.

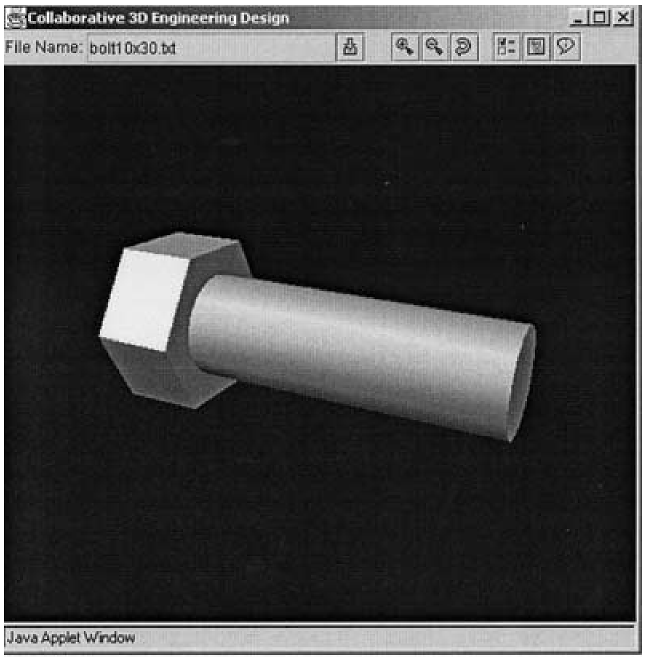


Figure 3. Geometric modeling environment.

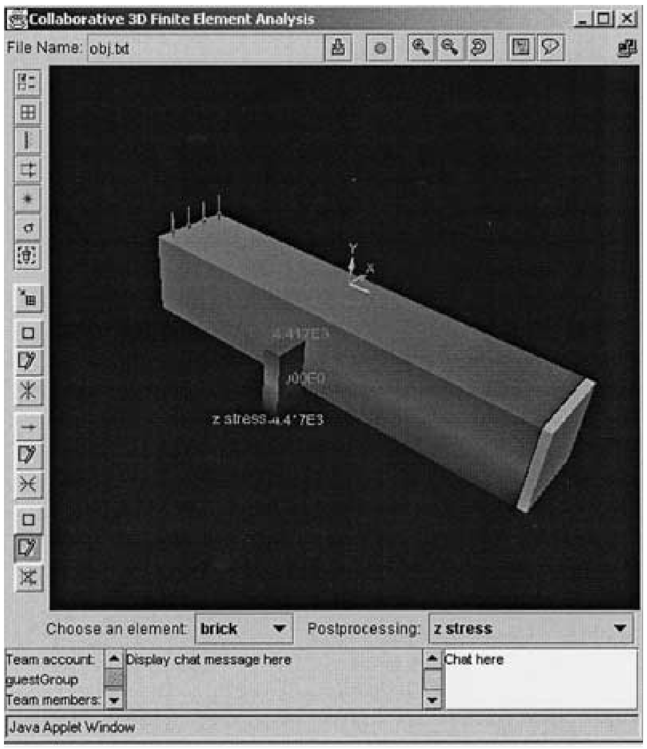


Figure 4. Color representation of *z* stresses.

features. It can import the geometric data created by the geometric modeling environment and display it in wireframe. Because the geometric data are stored in a file using the B-Rep data structures, the analysis environment reads in the data structures and converts them into internal representation of the geometry. The user can then specify the material properties and units for the imported geometric object. Automatic meshing was developed to prepare the data input for finite element analysis. The user can assign the number of elements associated with three edges to perform meshing (Figure 2). Two kinds of elements: a tetrahedral element and a hexahedral element were implemented. The user can select the preferred element.

After the meshing is over, boundary conditions, point forces, and/or distributed forces can be applied. Three buttons in the environment are grouped to deal with boundary conditions. First, the user depresses the 'Pick Quads' button, and he/she then picks desired quads. The selected quads become green. Second, the user clicks on the 'Boundary Condition' button, and a window pops up to let the user specify the degrees of freedom for selected quads. If the user chooses to click on the 'Enter' button, a colored block is placed on the top of the green quads to indicate that a boundary condition is applied on these quads. If the user wants to change the degree of freedom of the boundary condition, he/she can depress the 'Edit BC' button and then pick the colored block that represents the boundary condition. A window pops up to let the user modify the degrees of freedom. If the user decides to delete the generated boundary condition, he/she can depress the 'Delete BC' button, and then click on the desired colored block to delete it.

Two types of forces can be applied in the analysis application: point forces and distributed forces. The user depresses the 'Point Force' button, and then picks any node. A window pops up to let the user specify the value of the point force. If the user clicks on the 'Enter' button, a point force is created with the direction specified by its three components. Likewise, the user can use the two buttons below to edit and delete the point force. It is more complex to specify the distributed forces, and the procedure is similar to create the boundary condition. The user first depresses the 'Pick Quads' button, which is below the 'Delete Point Force' button, and then picks desired quads. The user next clicks on the 'Distributed Forces' button, and them picks desired quads. The user next clicks on the 'Distributed Forces' button, and a window pops up to allow the user to define the direction and value of distributed force. If the user clicks on the 'Enter' button, the distributed force is generated, which is located in the middle of the quads. Similar to the point force, the user can edit or delete the distributed force.

The last step is to conduct computation and check the results visually (Figure 4) or read the result report. The computational results from the collaborative environment were within 1% of the results using ANSYS. Figure 5 demonstrates the stress computation of a retaining wall using this collaborative analysis environment.

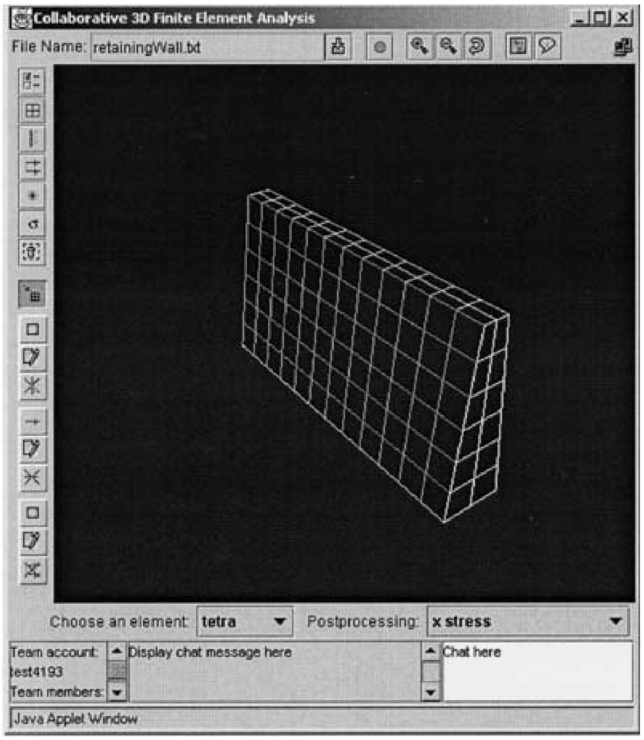


Figure 5. Analysis of a retaining wall.

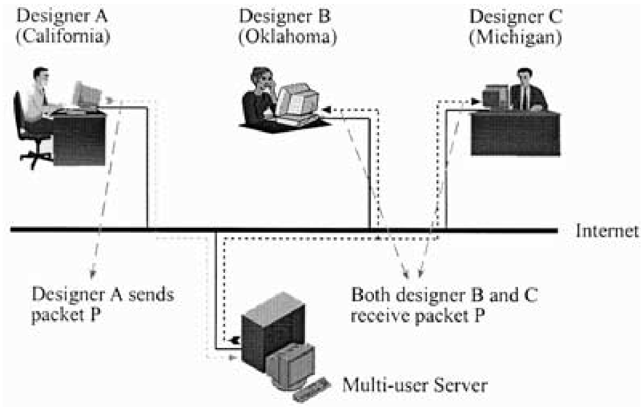


Figure 6. Communications among client-side applications and server.

To illustrate how distributed collaboration works in the engineering analysis environment, let us assume there are three engineers in an analysis team. Engineer A is in California and serves as a team coordinator; Engineer B is in Oklahoma; Engineer C is in Michigan (Figure 1). These three engineers first create individual user accounts on the environment. The team coordinator generates a group account, adds his team members to it, and informs them of its name. By providing the group account and individual user account, they can log into the environment and share the same analysis with team members. Different teams have different group accounts so that only members within the same team can conduct analysis collaboratively. An internal database of the multi-user server keeps track of group accounts, member accounts, and active connections to the server. The client-side applets talk to the server, and the server routes their messages to other client-side applets with active connections in the same group so that cross talking among groups is avoided (Figure 6). A unique token is assigned to each group to avoid operational conflicts. For example, if a team member wants to rotate the model, his client program checks whether other clients are taking the token. If no one is holding the token, his client program grabs the token and allows his rotational actions. If the token is not available for the time being, rotational actions are not allowed.

Engineer A and B log into the environment and start the applets to conduct analysis. Through a chat room, Engineer A discusses with B which file they need to work together. Engineer B then loads the target geometric data file into her applet, and the same file is loaded immediately on the applet of Engineer A. Engineer A decides to assign the number of elements along three edges and perform automatic meshing to the geometric object. The meshing is propagated to the applet of Engineer B, and Engineer B sees the meshes. However, Engineer B thinks the meshing is not good and discusses it with Engineer A. Engineer B then deletes the previous meshing and conducts her meshing. The deletion and new meshing are propagated back to the applet of Engineer A. Engineer C then logs in the environment and starts the applet. As they all log into the same team account, Engineer C immediately sees the work-in-progress from the other two engineers. After they finish working on the boundary conditions and forces, one of them starts the computation and the computation command is propagated to the other two engineers' applets. All of them obtain the same computational results, and they are able to discuss the results by visually sharing the same viewpoint. For example, say, Engineer C wants to discuss von Mises stress with the other two engineers. He selects the von Mises distribution on his applet. The other two engineers instantly see the same demonstration on their own screens. If Engineer C wants to investigate the analysis by himself, he can temporarily leave the team. In this case, any action by other two engineers will not be propagated to him, and his behavior will not affect the analysis conducted by the other two engineers. Engineer C is disconnected from the other two engineers. If Engineer C joins the team again, he will lose whatever is on his applet and obtain a copy of the work-in-progress of the other two engineers.

It should be noted that this environment adopted a strategy different from conventional architectures. Current collaborative design environments usually use classical server/client architectures and have powerful geometric modeling kernels running on the server. The functions of clients are quite limited and mainly for displaying the models. This architecture therefore imposes high traffic between the server and clients. This project proposed and implemented a different strategy, which can significantly reduce the network traffic between the server and clients. Although the server/client architecture is still used, the major functionality is moved from the server to the clients. The server only takes care of the communication. The clients have the full capabilities to conduct analysis and present the results. Only a small amount of data is transmitted over the Internet in each operation. This is because the transmitted data are not the computational results, but related multiple commands.

Since the engineering analysis environment is just a prototype to demonstrate the concept of collaborative engineering analysis, not a full, commercial finite element program, there exist a number of limitations. For example, it can automesh only geometric objects with six faces since automeshing of arbitrary geometric objects was not implemented.

4. Design of the Collaborative Environment

In addition to the implementation of the finite element method, the design of Internet-based distributed collaboration also plays a vital role in the environment. This is because the environment is not a stand-alone application; rather, it is an application with communication capabilities. The general communication strategy is to share actions between geographically dispersed users. These actions include importing geometric files, meshing the objects, applying boundary conditions and forces, conducting computation, rotating the objects, zooming in on or zooming out from the objects, and viewing the results. When a user initiates one of the actions, that action will be propagated to the remote users. Therefore, all of the remote users in the same working team will have the same object, same view, and same results.

To design an efficient application with distributed collaboration, numerous situations need to be considered. First, it is important to find a consistent and convenient approach to retrieve data due to the large amount of data transmitted through the Internet. Object serialization is an excellent solution where the sender encapsulates the data into a serializable object and sends it. The receivers get the serializable object and cast it back to the original object. The receivers can then retrieve the data. This is particularly convenient because the data items have names associated with them.

Second, the finite element method has to deal with a large amount of data. Thus, only the action commands are shared, not the generated data. The generated data are usually a large amount, while the data for commands are small. This method can minimize the data that need to be transmitted through the Internet; hence it reduces the traffic imposed on the Internet and increases the responsiveness of the environment. The commands are encapsulated into serializable objects and transmitted over the Internet. When the receivers get them and retrieve the data, they then take actions based on these rule-based commands.

Third, multiple copies of information may be sent to the new user when the existing users detect the join action. To avoid this, a unique token is used for the working team. If one user is holding the token, other users cannot hold it. Therefore, only the user holding the token can send a copy of information to the new user. Other users have to ignore the join action even though they have detected it.

Fourth, a user may leave the working team and rejoin it. When the user leaves the working team, that person does not send or receive any message from other users. The individual can work independently on geometric objects. However, if the person decides to rejoin the working team, his objects will be lost when he receives the current copy of the working object from the team. This is important because only one copy of information can exist within the team.

Fifth, each user may create new objects, such as boundary conditions and forces. These new objects can be easily identified in the creator's local machine, though not in the remote users' machines. To solve this problem, a globally unique ID is assigned to the generated object by the creator's machine and then propagated to the object generated by remote user's machines. Hence the generated object in each machine has the same unique ID. When the user needs to edit or delete the generated objects, the unique IDs are used to distinguish objects.

5. Conclusion

This paper discusses the design and development of an engineering analysis framework that allows remote users to conduct finite element analysis collaboratively over the Internet. Java and Java 3D were chosen to implement the working prototype due to their platform-independence and networking capabilities. Different from classical server/client models, the environment adopts a strategy in which the client-side application has full analysis capabilities while the multiuser server only manages communication. The commands are transmitted over the Internet instead of the generated data, and network traffic associated with the collaboration is minimized. Further work is being done to implement a generic meshing strategy, support more elements, and add voice communication.

Acknowledgment

We acknowledge the financial support in part from the National Science Foundation, Grant No. EEC-9872505.

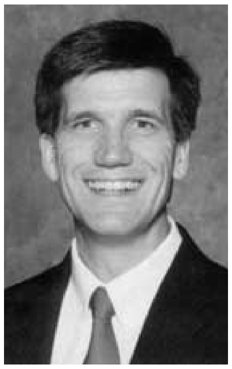
References

1. Sun, Q. and Gramoll, K. (2001). Internet-based Distributed Collaborative Environment for Engineering Education and Design, *2001 ASEE Annual Conference Proceedings*, Albuquerque, NM.
2. Sun, Q. and Gramoll, K. (2002). Distributed Collaborative Environment for Geometric Modeling, *Engineering Design Graphics Journal*, **66**(1): 24–31.
3. Prasad, B. (1996). *Concurrent Engineering Fundamentals*, Prentice Hall PTR, Upper Saddle River, New Jersey.
4. Mills, A. (1998). *Collaborative Engineering and the Internet*, Society of Manufacturing Engineers, Dearborn, MI.
5. Gates, B. (25 June 2001). Microsoft. NET Today, *eWeek*, **18**.
6. Pahng, G., Senin, S. and Wallace, D. (1997). Modeling and Evaluation of Product Design Problems in a Distributed Design Environment, 1997 ASME Design Engineering Technical Conferences, Sacramento, California, 14–17 September.
7. Case, M. and Lu, S. (1996). Discourse Model for Collaborative Design, *Computer-Aided Design*, **28**(5): 333–345.
8. Lee, J., Kim, H. and Han, S. (1999). Web-enabled Feature-based Modeling in a Distributed Design Environment, 1999 ASME Design Engineering Technical Conference, Las Vegas, Nevada, 12–15 September.
9. Smith, C. and Wright, P. (1997). Cybercut: A Networked Manufacturing Service, First International Conference on Managing Enterprises-Stakeholders, Engineering, Logistics and Achievement, Loughborough University, UK, 22–24 July.
10. Mori, T. and Cutkosky, M. (1998). Agent-based Collaborative Design of Parts in Assembly, 1998 ASME Design Engineering Technical Conferences, Atlanta, Georgia, 13–16 September.
11. PTC 2001, http://www.ptc.com.
12. Cocreate 2001, http://www.cocreate.com.
13. Vinacua, A. (2001). Issues in Distributed CAD, http://deslab.mit.edu/DesignLab/dicpm/position/vinacua.html.
14. Rossignnac, J. (2001). Collaborative Design and Visualization, http://deslab.mit.edu/DesignLab/dicpm/position/rossingnac.html.
15. Loosley, C. (June 1998). Designing Distributed Applications, *PC Magazine*, Ziff Davis Media, Inc., New York.
16. Salustri, F. (1998). Issues in Internet-Enabled CAE. In: Rosen, M.A., Nayler, D. and Kawall, J.G. (eds), *Proceedings of 1998 CSME Forum*, pp. 21–27.
17. Peterson, L. and Davie, B. (2000). *Computer Networks – A Systems Approach*, Morgan Kaufmann Publishers, San Francisco, California.
18. Olsen, J. (October 2000). CAD Embraces the Internet, *PC Magazine*, Ziff Davis Media, Inc., New York.

Qiuli Sun

Qiuli Sun received his Ph.D. in Mechanical Engineering at the University of Oklahoma in 2001. He worked at Shanghai Automotive Industry Technology Center (SAITC) for about three years as an automotive engineer after receiving his B.S. in Mechanical Engineering and M.S. in Automotive Engineering from Tongji University, Shanghai, China. At SAITC, he analyzed the performance of vehicles and conducted finite element analysis. At the University of Oklahoma, he developed a collaborative Internet-based virtual world for engineering education. His research interests include finite element analysis, dynamics analysis, vehicle dynamics, and Internet-based engineering education. He is currently working as a senior developer at Virginia-based ExploreLearing.

Kurt Gramoll



Kurt Gramoll is the Hughes Centennial Professor of Engineering and Director of the Engineering Media Lab at the University of Oklahoma. He has developed and published CDs and web-based sites for engineering education, K-12 instruction, and training in industry. He has started two multimedia companies for the development and distribution of technical electronic media. Dr. Gramoll received his B.S. degree in Civil Engineering and M.S. degree in Mechanical Engineering, both from the University of Utah. He received his Ph.D. in Engineering Science and Mechanics from Virginia Tech. Previously, he has taught at University of Memphis and Georgia Tech.