Development of a collaborative multi-user engineering design graphics collaboration tool

Carlos R. Morales, Michael Goldenberg
Purdue University

Abstract
The ability to collaboratively work on engineering graphics is of a great advantage. This paper details the development of a multi-user multimedia tool specifically designed to enhance collaboration in the engineering design graphics arena. The system provides users with the capabilities to collaboratively explore 3D environments, video-teleconference, and share applications files.

The system provides a centralized application that combines traditional CD-ROM multimedia tools with web tools to provide a media-rich collaborative environment. Specifically, the system is built around Macromedia Director and Microsoft's Active Server Pages. Macromedia's Shockwave Multi-user server is used as a conduit for synchronous communications, and ASP pages are used for the storage and administration of shared spaces. Microsoft's Media Services and Active-X components are used to enhance the collaborative capabilities of the system.

The system also address on-site collaboration through the integration of an intelligent mechanism capable of identifying users accessing the system from wireless PDA's and delivering alternate content through IEEE 802.11b Wireless LAN.

Introduction
As the complexity of projects increases, the need for collaboration tools has increased. Not only do workers need the ability to collaborate on their actual designs, but also on the project-management issues that accompany any large scale design endeavor. Most developers of engineering design software have answered this problem by providing some degree of collaborative tools within their packages. It is not uncommon to find an array of collaborative tools in today's major engineering design packages. A quick survey of the toolsets available in packages such as CATIA, ProEngineer, SolidEdge/SolidWorks, Autodesk Mechanical Desktop, etc. reveals a wide range of features ranging from web-viewer to project-management [1,2,3,4].

The major weakness with these collaborative tools is that most offer a large degree of option for their specific vendor's tool, but little or no support for other vendors' software. These packages work great when two or more workers will be collaborating on a project using software from the same vendor or from vendors "preferred" by the software of their choice. Typically, this has been dependent on the ability of the software used by the worker to import data from other vendors.

Some third-party developers have attempted to bridge this gap among tools by providing their own collaborative tools. 3G.web.decisions allows designers to not only view design from a variety of different engineering-design packages, but also make changes on-line [5]. Actify also has a variety of collaborative tools that allow web based collaboration during the design phase using native data from just about any imaginable engineering-design software [6].

These collaborative tools fall short in collaborative efforts that fall outside of the primary purpose of the tool. They do not address collaboration across a wide-variety of personnel in most organization. The above tools would not provide the needed support to allow collaboration in the following situation. Lets assume that we had one engineer working in CATIA, another engineer using an in-house proprietary finite analysis tool written in FORTRAN, a marketing person working in QuarkXPress responsible for putting together a catalog of the parts, a video specialist working in Adobe After Effects responsible for putting together a marketing video, and a project manager. While the majority of existing tools would allow people in the above situation to export/import files required for them to complete their job, none of the existing tools would allow this to occur in a single coherent environment.

It is this need that our collaboration tool attempts to fill. The goal of our system to facilitate collaboration under any environment and any condition by providing an open-architecture environment that allows the user to select the features they need. Instead of trying to recreate the existing tools already in existence inside of the major engineering design packages, our system allows the user to leverage those tools for their specific situation. By not dictating a specific set of tools and providing an open-architecture the system provides collaborative features that meet most everyone's needs.

Background

Our collaborative tool, Engi-Share, shares roots with an in-house distance-learning tool developed at the Computer Graphics Technology Department at Purdue University called the Unified Multimedia Delivery System (UMDS). The UMDS strives to provide an open-architecture environment for conducting distance learning through the integration of COM objects, ASP, and Macromedia Director.[7] Engi-Share also draws from a wireless instructional media delivery mechanism developed at Purdue to augment the UMDS. [8]

The authors found that the engine developed for the UMDS could be adapted for any situation that requires on-line collaboration including engineering-design collaboration. Both situations required the sharing of application data. Thus, the application sharing features could remain relatively the same. Classroom specific features such as test and lecture administration were deleted because theses would not be needed. Project management features had to be added because they did not exist. The user interface and programming interfaces were also adapted to the current task.

Table 1, below reflects the current features of the system.

Engi-Share Features
• Application sharing of any Win32 program including CAD and FEA packages • Database driven contact manager • Timeline • File check-in/check-out • File sharing • Integrated file translation • Video teleconferencing (Syncrhonous) • Audio teleconferencing • Whiteboard • Text based chat • Semi-synchronous high quality video presentation at VHS or better quality • Collaborative web browsing

Table1. ENGI-SHARE Features

To achieve this functionality, Macromedia Director 8 and embedded Active-X components were used. Director provided the interface and media delivery capabilities, NetLingo and the Macromedia Multiuser Server provided the TCP/IP calls necessary for creating a message loop between the client and server applications, Active-X components provided application sharing and allowed the client stations to receive live video streams, and Microsoft Media Services provided the ability to send a live multicast stream.[9,10]

Crucial to the development for the ENGI-SHARE was the establishment of a control mechanism among the client applications. By using Net Lingo to create a chat channel among the client applications and using Macromedia Multiuser Server as a hub to reflect messages, the author established an inter-application message loop. Event handlers were then written to intercept events sent through this mechanism. This provided the needed control among the individual program pieces.

While not a critical part of an engineering-design collaboration tool, the authors decided to put some emphasis on the video exchange capabilities of their system. Not only did they want the system to excel at delivering synchronous video presentations over high-bandwidth connections, but also to low-bandwidth clients. To synchronize the video presentations among the clients and the server three mechanisms were employed. First, the server was programmed to push a new database file onto the client applications numerous times throughout the day. Upon receipt, the client application compares the database version of the presentations on the database with the versions on the local hard drive. If discrepancies arise, the client stations download the new files. This mechanism insures that the clients have the newest presentations most of the time. For last minute changes, the clients are programmed to check their database with the master server database fifteen minutes before presentation starts. If the versions differ, the client pulls the newest presentation off the server. The retrieval of the video files is accomplished through NetLingo's FTP functions and by communication between the client and the server through the chat based virtual message loop [11]. DataGrip was used to access the Microsoft Access database from within Director. An option to manually force an update was also placed on all of the clients.

At this point the application correctly handled video presentations delivered in advance to the clients. The system could adjust to any changes in video as long as these changes occurred within 15 minutes before presentation-time. Now it was time to implement a mechanism for live video presentations. A message handler was programmed into the clients allowing the server to switch to a live video feed at any point. After experimenting with a few alternatives, the author decided to use Microsoft's Media Server to multicast a video stream. An embedded MS Media Player Active-X control allowed the author to receive video within the Director application. The Macromedia Active-X controller allowed the author to embed Active-X objects within the Director application without having to program the COM API directly. By writing a few event handlers and accessing a few provided methods, the author was able to harness the power of these applications from within the Director 7 ENGI-SHARE.

Microsoft's Media Services provided the means for streaming very high quality video from the server to the clients. Provisions for multicasting and unicasting made the use of this technology very attractive. By setting the up the system to multicast, the presenter was able to send a very high quality MPEG-4 stream from the server to a virtually unlimited numbers of clients. Rates as high as 3.25 megabits per seconds can achieve decent full-screen full-motion video. The decision to utilize Microsoft's Media Services for video transmission necessitates additional hardware on the server side. Two additional machines were needed. One machine housed a video encoding card and ran Microsoft Media Encoder. This machine was dedicated to digitizing the video and encoding it into MPEG-4 during live video transmission. A second machine ran Windows NT Server 4.0, Microsoft Media Server, and Information Server 4.0. This machine was responsible for taking the stream coming from the encoder machine and multicasting it.

The server could instruct the clients to show full screen MPEG-I video from their own hard drive or CD-ROM, or to accept a very high quality MPEG-4 live video stream. A mechanism for live video and/or audio from the clients back to the server was still needed. The author elected to use NetMeeting for this purpose because it could be deployed on the clients without requiring multiple machines and it could be embedded within the host Director application. An additional benefit was that video transmitted by NetMeeting is less bandwidth intensive than that generated by Microsoft's Media Services.

The NetMeeting COM object provided a large amount of functionality to the UMDS. Microsoft not only provides NetMeeting as an application and an ActiveX object that can be fully embedded into an application, but also provides access to functions that allows us to selectively use features provided by NetMeeting. This allowed us implement video audio conferencing easily. The negative aspect of using this approach is that for multi-point video teleconferencing the users require access to a MS Exchange Collaboration server which provides access to a multi-point control unit.

Application sharing was also achieved through the integration of selected NetMeeting functions into our Engi-Share. This approach allows the user to use any engineering design package they would like to use. In fact, it allows engineers to even collaborate with people who do not own the package they are using. It makes all participants, users of a single application. NetMeeting also provided application sharing and collaboration, file transfer, and a whiteboard.

The time-line and contact modules were implemented by using Active Server Pages to deliver the content to the Director application. An embedded Microsoft Web Browser Active-X component allowed the clients to access the content without having to spawn an external browser. The reliance on ASP necessitated the use a PC running Windows NT Server 4 and Microsoft Information Server 4.0. A benefit of using this technology was that it made all of the server side technology, such as server-side database access or banner rotation, available to our application. Using the embedded web browser allowed the author to control the content the user sees. Because the Director application retrieves the web content and presents the information to the user transparently through its own interface, the user never gets a chance to “browse the web.”

On the server, the participant’s responses were routed to an Access database on the IIS server by creating an Active-X Data Objects connection. Using this mechanism IIS is able to interface with any ODBC compliant database.

To account for on-site collaboration, the authors developed a scaled down version of the Engi-Share tool, built entirely as a web application using ASP. By checking the USER_AGENT of any incoming connection, the application was able to detect that the users were accessing the system using a IE on a PocketPC PDA. Based on this, the content could be altered for the limited capabilities of the device. The actual delivery of the content occurred over a IEEE 802.11b wireless networking connection. Each of the testing laptops and PocketPC’s was outfitted with one of these cards. A desktop system was outfitted with an IEEE 802.11b access hub and a standard 100-megabit NIC, which connected to the network infrastructure. This machine served a bridge between the Internet and the wireless devices. Each card provides a theoretical bandwidth of 11 Megabits per second, which was more than sufficient for delivering full-screen full-motion video, but only had a range of one hundred fifty feet. Multiple access ports can be placed throughout an organization to provide a wireless access area beyond the 150 feet restriction of a single access hub. Within the confines of the development phase of this project a single access hub was used. It worked as expected. The author found the maximum sustained bandwidth to top out at approximately 7.6 megabits per second.

The interface also had to facilitate collaboration. We decided to devote most of the screen real estate to the application being shared by the user. Figure 1 shows the Engi-Share interface.

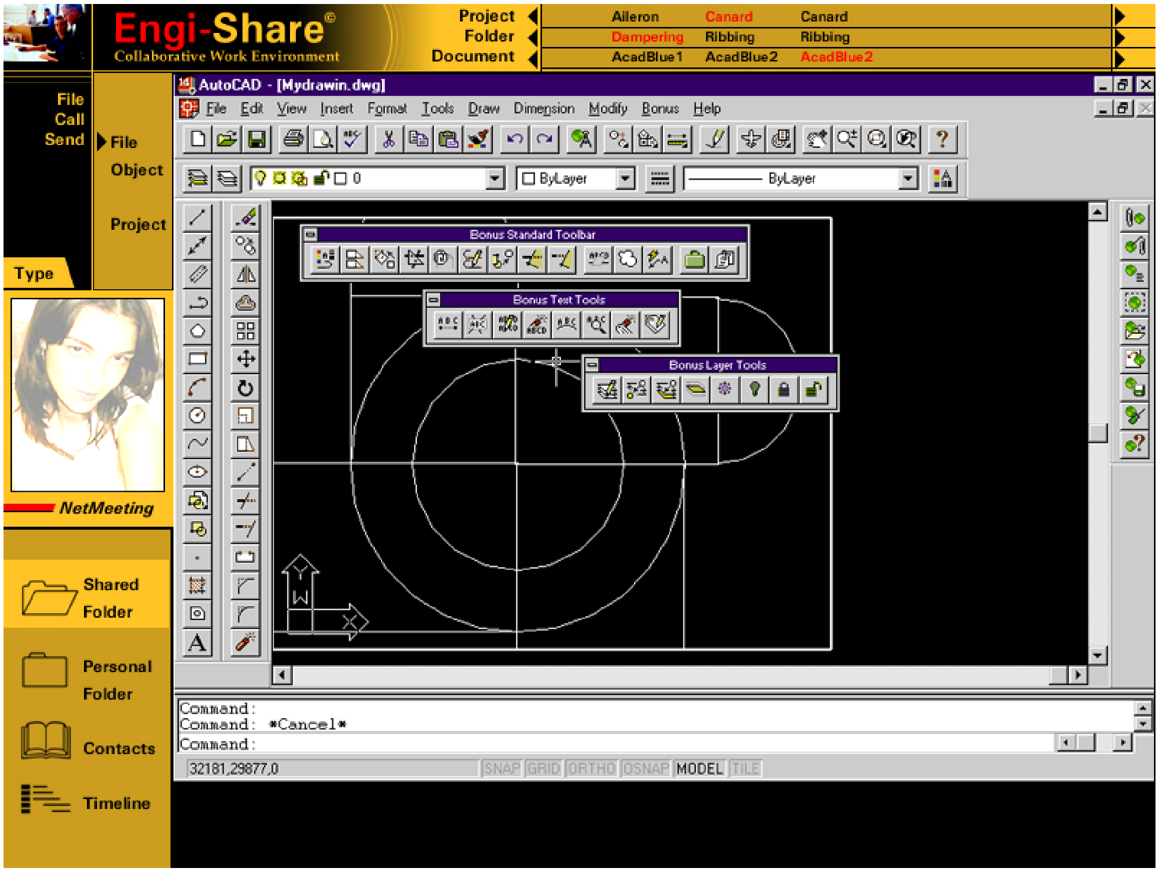

Figure 1. ENGI-SHARE Interface

Conclusion

By relying on an open-architecture of loosely coupled components the Engi-Share system achieves its goal of providing an interface that allows engineering-design collaboration in most situations. The system not only allows workers to collaboratively view the same engineering drawing, but also make changes on those drawings, regardless of the CAD or engineering-design software installed on the participant's machine. Only the host of the collaboration must have software installed on their machine. And because the participants are using the native system on which their designs were developed they have the full power of that environment available to them.

In addition, the Engi-Share system provides a way for workers from different departments to collaborate on the same project by providing a mechanism that allows the sharing of almost any software package. In this scenario a marketing person and an engineer can share their respective applications and collaborate with each other. Our system provides the translation meachanism to allow them to share most of their files. If the system cannot translate their files, its open architecture provides the flexibility for expansion.

The system also provides less engineering-design specific collaborative tools such as file transfers, file check-in/check-out, video/audio conferencing, timeline management, white boarding, etc. These features, while not critical, enhance the collaborative experience.

The next phase in the development of the Engi-Share is to field test it in a variety of engineering design settings. Improvements in the area of security are also needed. As the system currently stands some level of secure communication is possible through the embedded web browser using 128-bit SSL encryption. This level of security needs to be implemented across the entire application. In the end, the system does provide the level of functionality one would expect from the first version of a software solution.

Bibliography

[1] Mechanical Desktop product information (2002) AutoDesk Corporation. [On-line] . Available: http://www3.autodesk.com/adsk/section/0,,618123-123112,00.html.

[2] CATIA product information (2002) Dessault Corporation. [On-line] . Available: http://www.catia.com/

[3] Pro/ENGINEER product information (2002) PTC Corporation. [On-line] . Available: http://www.ptc.com/

[4] SolidWorks product information (2002) SolidWorks Corporation. [On-line] . Available: http://www.solidworks.com/

[5] Greco, Joe. CADENCEWeb (2001). CMP Publishing. [On-line]. Available: http://www.cadenceweb.com/2001/0801/mcad0801.html

[6] 3Dview and SpinFire product information (2001) Actify Corporation. [On-line] . Available: http://www.actify.com/v2/index.htm

[7] Morales, C., & Meador, S. (2000). Trial results from a custom distance learning system based on hybridization of cd-rom, web, and active-x component technology. Proceedings of the Information Visualization Conference IV'2000 (pp. 39-43). IEEE Computer Society, Los Alamitos, CA.

[8] Morales, C. (2001). A method of delivering dynamic hyperlinked streaming instructional content over a wireless environment through the use of active server pages and xml technology. 2001 Annual Conference Proceedings . American Society for Engineering Education, Albuquerque, NM.

[9] Active-X specifications. (2002) Microsoft Corporation. [On-line] . Available: http://www.microsoft.com/com/tech/activex.asp

[10] Media services developer Information. (2002) Microsoft Corporation. [On-line] . Available: http://www.microsoft.com/windows/windowsmedia/en/developers/

[11] Director user's manual. Macromedia Press, (2001) San Francisco, CA, 2001.

CARLOS R. MORALES

Carlos R. Morales is an assistant professor of computer graphics at Purdue University at Kokomo. He holds a BA in Telecommunications and an MS Ed. in Curriculum and Instruction. Prior to working at Purdue University, Carlos worked as a

Technical Director. Some of his clients have included Microsoft, Chicago Bulls Organization, First Alert and Brach's Candies. His research interest includes distance learning, animation, and multimedia development. He can be reached at crmorales@tech.purdue.edu

MICHAEL GOLDENBERG
Michael is a graduate student in the department of Computer Graphics Technology at Purdue University. He holds a B.S. in Technology. He can be reached at goldenbe@purdue.edu.