Implementation of a multi-user on-line geometry collaboration tool

Carlos R. Morales, Ronald Glotzbach, Dioselin Gonzalez
Purdue University, Knoy Hall, Room 363, West Lafayette, IN, 47907

Abstract
While the majority of CAD systems today have the ability to engage in online collaboration, most systems emphasize engineer-to-engineer collaboration and have little features for creating highly polished interactive presentations for general collaboration. Some of the systems do have web centric collaboration tools, which allow general collaboration, but like all browser-based web-systems, they lack the level of interaction that can be achieved with a dedicated interactive multimedia tool. The author presents a methodology for creating fully interactive 3D multi-user collaborative presentations through the extraction of 3D geometrical data from a CAD source. The data is then assembled and scripted using shockwave 3D studio. The end result is an on-line experience with accurate models and engaging multimedia content.

Introduction

On-line collaboration is extremely important. Not only is it necessary for engineers to collaborate with each other, but also in most commercial enterprises there is a need for collaboration with personnel who are not engineers such as non-technical management, marketing & financial personnel, and even customers.

In general, engineer-to-engineer collaboration problems have been the focus in the creation of collaboration tools. Most CAD systems today either have a web based collaboration module or can facilitate collaboration from within the software [1][2]. There are even systems that specialize in inter-application collaboration. Alibre Design makes a CAD tool that allows engineers to collaborate using different CAD packages [3].

In non engineer-to-engineer scenarios these tools suffer both from a technical perspective and from a usability perspective. This can be attributed to the fact that these tools have been adapted from engineering tools and do not focus on media presentation and delivery.

Most of these collaboration systems can be placed into one of two categories. In the first category we have web-centric systems. These systems allow users to assign roles, share drawings, use messaging services, etc. While the web centric nature of these tools makes them very accessible, it is also a shortcoming. All of these systems are plagued by the limited interactivity of the browser in which they are displayed. Technologies like client side DHTML, JavaScript, server-side technologies like ASP, JSP, PHP can all be leveraged to create a highly interactive experience. Yet, even at its best these technologies combined cannot provide the level of interactivity that a fully interactive multimedia package can provide.

In the second category we find collaboration systems that are built into the CAD systems. These are great for the engineering design process, but not for collaboration with non-engineers because they force everyone to have access to the host CAD package. Most CAD systems also lack the media manipulation tools inherent in a multimedia development package.

The traditional view and concept of collaboration in business has evolved during the past years. Integrated Collaborative Environments (ICE) that have been used to organize meetings and email and to build custom applications are now being complemented –or substituted– by other collaboration applications, such as data-conferencing and instant messaging software which provide missing functionality from ICE products and a superior ease of use and user control [3.1].

Vendors of workflow applications are joining forces with vendors of collaborative applications to create rich applications with structured collaborative features that don't force the user to work in one application to learn a fact and then use another collaborative application to convey that fact or take action based on that fact [3.1]. The term Contextual Collaboration has been introduced by IDC to describe processes that introduce collaboration to business while making it easier to users [3.1].

Project goal
With this system the author wanted to go beyond traditional ICE applications towards Contextual Collaboration to enhance the collaborative experience. The focus of the project became the development a system that could be used for sharing geometric data that originated in a CAD system among non-engineering participants in a media rich environment.

A plausible use for this tool could be the following:

> A marketing person from a company that manufactures pistons would like to show his products to a prospective client. He does not only want to show off the product, but also to receive feedback on it. He would like to show a 3D model rendered in real-time, some textual documentation such as performance tests, and video clips of parts in both testing and operation. In addition he would like to give the client the ability to suggest changes to the 3D model if need be or even the ability to render 3D visualizations.

In short, the system would need not only to allow the user to show geometric data, but also to orchestrate the display of auxiliary information such as video, text, and test data.

Development
To accomplish this task the author selected Macromedia Director MX as the front-end development environment because of its extensive media manipulation tools, its Internet savvy tools, its extensible 3D engine, and its ability to interface with COM objects. The back-end development was accomplished using Microsoft's Internet Information Server and SQL Server. The main interface into the sharing application was built using Director and deployed as Shockwave [5].

Interactions

The system allowed for multiple levels of interactions. At its basic level it allowed participants to display to each other 3D models, video, text, and audio. From a developer's perspective there were two scenarios that needed to be considered.

First the author needed to account for situations in which the user would like to just throw items on the screen at a whim. In this scenario nothing is predetermined. It is like someone having a phone conversation. From an implementation point-of-view this was easy to accomplish. All of the client machines created a chat connection through the Shockwave Multiuser Server. Each client had the capabilities to send commands to the server, which then broadcast those commands to everyone connected to it. Each client also had a set of callback handlers that received the broadcasted events and reacted accordingly. In other words this created a virtual event-driven messaging hierarchy among the participants. It worked well.

In the second scenario, users would be allowed to call on pre-made interactions. For example, the user might want to show a particular 3D model a particular piston for 20-seconds, followed by a 2-minute video clip of that piston being tested, and finally followed by a pdf file of the test data. If the user had to do this many times, it would be useful if the systems would allow him to have this pre-scripted. At run time, the user would just call on the script and all of these actions would occur. Employing simple parses and an interpreter accomplished this task. The commands were saved in an XML file and stored on a central server.

Three dimensional graphics

A central portion of the application revolved around utilizing and displaying 3D models. Director is able to render 3D scenes saved in Shockwave 3D (SW3) format using OpenGL or DirectX. Most CAD packages are unable to write in this format directly. To minimize the number of steps imposed on the users of the systems, the author decided to write importers into director that could read any common file formats such as OBJ and DXF. Exporters for 3DS Max, Maya, and Lightwave 3D provided SWD files from those respective packages.

Director's 3D environment is extremely powerful. It allows scripting of any aspect of the 3D world including vertices, lights, shaders, etc. Thus not only could models be imported but they could also be animated on the fly.

In authoring mode, it is relatively easy to issue commands using Lingo to drive changes to the 3D environment. At run-time, the user cannot issue Lingo commands directly. Thus a conduit needed to be built to allow users to issue commands. This was accomplished by building a simple parser and interpreter, which accepted proprietary commands and issued the appropriate commands to the Director run-time engine.

Three types of 3D interactions were considered. First, the author accounted for scenarios in which the users would want to display pre-made animations or changes to geometry.

This was accomplished by maintaining the geometric data separate from the animation data. At run-rime the animation data would be loaded from SW3 files that were local, on the server, or on the other participant's machines. Then it would be cloned into the 3D environment. Using this methodology was beneficial because it allow pre-scripted changes to the 3D environment to be called at any time and from any source.

The author also considered the scenario in which the participants wanted to apply a simulation or data from a simulation to the 3D world. In this case the most critical thing was that the models would need to accept data from a user defined simulation constructed at run-time and behave appropriately across all of the participant's machines. This proved to be challenging.

In order for this to work, the author needed to allow users the capability of defining mathematically driven processes and mapping the resulting data to the vertices of the 3D model. This was also accomplished via a scripting mechanism. Finally, there would need to be a way to coordinate the display of the results across all of the participants. This was done by running the simulation on a single machine, sending the results to all of the client machines, and finally using TimeOutObjects in Director. TimeOutObjects allow an event to be run at specific intervals using real-time values as opposed to relying on the processing speed of the machine. Thus all of the machines involved would stay synchronized regardless of the speed of machines involved. We did not account for line latency.

Video
Video was integrated into the application using three mechanisms. The author wanted to provide users of the tools with as many possibilities for using video as possible. Thus, three scenarios were considered.

In the first scenario, users want to provide prerecorded videos in non-streaming formats such as AVI or QuickTime. To accommodate this the author used NetLingo's asynchronous ImportFileInto command to download the media in the background and then present it at the appropriate time. Because the video would be in a non-streaming format, it would have to be downloaded in its entirety prior to using it. On a slow connection this delay could be considerable. Thus, downloading the file in the background was ideal. The user's application also was coded to inform the originator of the collaboration with the status of any media file downloads. This was necessary because the originator of the collaboration would need to know when it was safe to attempt to show the file video to the user.

In the second scenario, users of the system would be employing streaming video. The system would need to function with the three main video streaming formats in the marketplace, which are QuickTime, Real media, and Windows Media. Director MX has the ability to receive both streaming QuickTime and Real Media, but not Windows Media. To receive Windows Media the author embedded a Windows Media Player COM object within Director. This did have a negative aspect on the tool. COM is a Microsoft only technology. Thus the tool would only be compatible with Windows Media when running on a Windows system. On a Macintosh, users could only receive Real Media and QuickTime.

In the third scenario, users of the system would need the ability to stream live video. There are numerous solutions available for streaming live video. Some systems such as Microsoft's Windows Media Server can stream video at up to HDTV quality, but require three dedicated machines to operate. The author decided to employ a solution that could be implemented with a single machine. 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 and audio conferencing. 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 [6].

An unexpected benefit of utilizing the NetMeeting embedded object was that it provided the application with the added functionality of a whiteboard, application sharing, and peer-to-peer file sharing. The application sharing was extremely beneficial because it allowed the user to actually work together in a program outside of the collaboration application.

The time-line and contact modules were implemented 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 server running Windows 2000 Server and Microsoft Information Server 5.0. 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.

Conclusion

Traditionally, on-line collaboration tools have been created by and for the use of technical personnel –Internet was intended primarily for scientists and engineers. But with the spread of existent technologies and the advent of new ones, other fields have realized the great potential and benefits of on-line collaboration. These collaboration tools for non-technical people require two additional elements: standard packages of hardware and software to take advantage of the connectivity, and "ease-of-use" that allows use on a regular basis by personnel without engineering background. [4]
The developed system works relatively well at enabling collaboration among non-engineers. By relying on an open-architecture of loosely coupled components and

providing the users with strong media presentation capabilities, the system provides a strong multimedia collaboration environment. The system also provides file transfers, 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 system is to field test it in a variety of settings. Improvements in the network latency area are needed. While the system insures that events take place on the client's machines at predetermined intervals independent of machine speed, there is no mechanism for insuring that those events reach the clients' machines at the same time. In the end, the system does provide the necessary level of functionality.

Bibliography

[1] Mechanical Desktop product information (2003) AutoDesk Corporation. [On-line] . Available: http://www3.autodesk.com/adsk/section/0,,618123-123112,00.html.

[2] SolidWorks product information (2003) SolidWorks Corporation. [On-line] . Available: http://www.solidworks.com/

[3] Alibre Design product information (2003) Alibre Corporation. [On-line] . Available: http://www.alibre.com

[3.1] CIO Tech Current (2002) Collaborative Tools. [On-line]. Available: http://www.cio.com/research/current/tools/

[4] Center for US – Mexican Studies (2002). Virtual Collaboration. [On-line]. Available: http://www.usmex.ucsd.edu/research/virtual_collaboration.html

[5] DirectorMX product information(2003) Macromedia Corporation [On-line] . Available: http://www.macromedia.com/director

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

CARLOS R. MORALES
Carlos R. Morales is an assistant professor of computer graphics at Purdue University. 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. His research interest includes distance learning, animation, and multimedia development. He can be reached at crmorales@tech.purdue.edu

RONALD GLOTZBACH
Ronald is a Visiting Assistant Professor for the Interactive Multimedia Development area in the Department of Computer Graphics Technology at Purdue University's West Lafayette campus. He earned his B.S. in Computer Graphics Technology and holds a M.S. in Technology. Related interests include dynamic content delivery methods and integration of varying media into highly technological solutions. Contact him at: rjglotzbach@tech.purdue.edu

DIOSELIN GONZALEZ
Dioselin Gonzalez: graduate student pursuing a master's degree in Computer Graphics Technology at Purdue University. She has a B.S. degree in Computer Science from Simon Bolivar University in

Venezuela and has worked as a software developer and consultant. Her primary area of interest is multimedia and computer graphics programming. E-mail: dagonzalez@tech.purdue.edu