6.837 Computer Graphics
Final Project Proposal

Prepared by:
Chris Avrich
Hai Ning
Henry Wong

Multi-user Networked Modeling Tool

Overview

Our proposal is for a graphics modeling tool in the context of a multi-user environment. The primary application will be in design fields where collaborative design takes place, for example, in architectural design.

Architectural design is often a tremendous collaboration process between design professionals. The successful communication of requirements and goals for a project is critical in determining the project's overall success, both in concept and detailed design. Most existing CAD packages only allow a single user to build an architecture model, which is not the optimal solution in a collaborative environment.

What we envision is to build an environment where multiple designers can log on to the system from different places and work in the same modeling space. They will be provided with an elegant GUI to interact with the model, but more importantly, when they add or move building blocks, the scene will be dynamically updated on each user's screen. When designer A makes a modification to a model, designer B will see these changes in real time. Each user should be able to view the model from whatever viewpoint they choose; facilities should also be provided to look at the model from other users' viewpoints, in order to help them gain insight into their partners' ideas.

Behind the scenes, we would like to have a central database to keep track of each change in the model. At some stage, the designers will come to a consensus, at which point they can save their work to the database. Later on, they can retrieve the previous model and continue their work. The database will not only save the final product, but it will also contain the whole design process, because it will log changes along the way. Also, we may implement a plugin chat box and/or whiteboard in the applet to enhance communication. Everything the users type or draw could be associated with their model-building process, and would be saved for later reference. This way, it will be a lot easier for designers to work together and to understand one another.

Our basic system architecture can be broken down into a client part and a server part. The server part would include the application server, which receives client requests to update the scene, and broadcasts this data back to all clients. It would also contain the database server, which stores and retrieves the design process information when the application server calls it to do so. The client side includes the interface for each individual user to interact with the modeling space, as well as the connectivity to the application server.

Java and VRML seem to be appropriate tools to build our system. Cosmo World, from SGI, is a VRML plugin for web browsers which provides an API called External Authoring Interface to Java (similar to the Open Inventor API). Of course, the obvious advantage to building the system as a Java applet is that users can access the system from anywhere, on any platform, without anything but a web browser and a VRML plugin. For the back-end database, we could leverage the JDBC to connect to a database server to store information permanently. Other than Java and VRML, we could have also chosen to implement the system with tools like OpenGL, Open Inventor, Java 3D, etc. Java 3D may also be a good candidate, since it is built on top of OpenGL, and may give us more low-level control than VRML or Open Inventor API.

Division of Labor

Although we will all work on every aspect of the project, our group plans to divide the work up so that each person focuses on one specific task.

Hai will focus on the networking and message logging portions of the project. We plan to implement a broadcast server that also logs events passed through it. For the initial phases of the project, before we implement the networked connectivity portion, he will be helping write a mock-up of the server that simply takes every event the client sends to it and passes it back to the client. After this is done he will focus on writing up the full server so that events are broadcast to multiple clients. Hai will also be helping Chris with the clientside database.

Henry will be focusing on writing the java interface to the program and on making the client connect with the database. He will also work in conjunction with Hai to implement the client end of the network connection. In the later stages of the project he will also be writing the code to handle shared text messages and/or a shared whiteboard between clients. He will also be helping Chris with the client portion of the database.

Chris will be working primarily on the VRML side of the project. He will be working on implementing the client copy of the database and on the interface between the database and the VRML. In particular, he will be working on implementing individual viewpoints into the VRML world for each client, and allowing the cameras to interact via the network. For example, this enables you to show someone else the camera angle that you are seeing.

Timeline

By Wednesday, November 3rd, we hope to have completed the design document for the project. This document will be a description of the specifications of each of the modules required, and how they will interact with one another. Before Checkpoint 2 on November 5th, we expect to have programmed at least the bare mock-up of our user interface in java.

By Checkpoint 3 on November 12th, we plan to have the limited version of our applet available, whose server will merely pass all client requests back to the client. This will perform much like a single-user version of the project; it will not implement the web server beyond the bare necessities, but it should behave much like the finished project would with only one user.

By Checkpoint 4 on November 19th, we will have a functional version of the client-server architecture with a simple VRML modeling engine. We will not concern ourselves with the more advanced features of the modeling, but will focus primarily on the interaction between users and the server's database functionality.

By November 26th, although there will be no checkpoint, we plan to freeze our fully-featured code. We will implement as many interesting features in the modeling as we can, and smooth out any other problems our project may have before then. We will then be able to begin work on our final report and final presentation.

Timeline Summary

11/3 Wed	Complete Design Document
11/5 Fri	Checkpoint 2: Interface Mock-up
11/12 Fri	Checkpoint 3: Finish Single-User Version
11/19 Fri	Checkpoint 4: Working Client-Server Architecture with Database

11/26 Fri	Code Freeze
12/3 Fri	Written Report

return