3-D Collaborative Multiuser Worlds for the Internet

Bruce Donald Campbell

Rensselaer Polytechnic University, University of Washington

ABSTRACT

3d Multi-User Collaborative Worlds For The Internet is a Masters project looking into the potential of using VRML 2 and Java to create multi-user worlds on the Internet to foster collaboration between participants. The author reviews the history of technology-enabled collaboration and then reviews the current technologies available for creating interactive 3D collaborative worlds using the Internet to connect participants.

The author then explains his own approach to creating a flexible architecture in which virtual participants can share rules, objects, and actions between themselves while collaborating in a virtual world. The author suggests the importance of rules negotiation and design participation to making collaborative worlds succeed. The author discusses a collaborative multi-user 3D world of his design and then provides results on the success of the world after having subjects spend time together in the world working towards a pre-defined objective.

PROBLEM STATEMENT

A collective vision for a shared, 3-dimensional, Internet accessible cyberspace is rapidly becoming a reality in Silicon Valley. Successful technology companies are coming together and creating standards to support a cross-platform, cross-server, shared cyberspace. Living Worlds is an example of such a standard. Living Worlds focuses on standard connections of a web browser to the Internet and the requirements of a server to communicate with a compatible browser in a standard manner. Many 3D web browsers have recently incorporated an external interface API that will allow others to write the scripts that make cyberspace come alive. Server technology is rapidly improving the shared behavior routines that allow one browser to see the effects of the actions of another cyberworld visitor. So, its time to start building the worlds this technology will support. How effective will we be able to communicate, educate, and entertain ourselves in cyberspace over an Internet connection? The question is a wide-open one. It is time to begin to build worlds and test out our abilities to collaborate in them.

PROBLEM SIGNIFICANCE

Collaborative, shared digital worlds have been created in the past. Many have been created on dedicated networks such as the military's DIS network, Japanese research lab networks, or university laboratory networks. The shared spaces were always created at great cost and with little opportunity to make them available to the mass public. The commercialization of the Internet and the advent of the World Wide Web have made public availability to computer networks a possibility. The Internet is not a reliable deliverer of information. The attempt to port yesterday's shared digital worlds have failed. These applications required reliable and rapid delivery of information. Today, the technology is being built from the bottom up to work with the inherent weaknesses of Internet information delivery. The goals may be the same as cyberspace projects of the past, but the delivery strategies are quite different.

We have an opportunity to give cyberspace access to millions of world citizens. Some of these people will be able to truly participate in a cybersociety where they have struggled to participate in our society to date. Many physically challenged individuals are not mobile. They can't easily run out and participate in a spontaneous societal event such as a political rally or sports team celebration. Even a typical world citizen can't physically get to where the educational and entertainment opportunities are available. Instead, they settle for struggling to find the best opportunity in their own neighborhoods. In too many neighborhoods the best opportunities are not near good enough. They are limited by the experience and knowledge of the neighbors that live there.

The technologies needed for cyberspace are falling into place. It is time to build cyberspace as it should be built to reach out to the needs of our society. It is everyone's responsibility to build it right. I want to do my part by understanding the technology, trying it out, and providing feedback through a significant project/thesis paper. The Living Worlds standard addresses networking, user interface, application programming interface, and avatar representation issues. I will be learning specifics that will have broad, transferable teachings for me.

1 INTRODUCTION

A collective vision for a shared, three-dimensional, Internet accessible cyberspace is rapidly becoming a reality in Silicon Valley. Successful technology companies are coming together and creating standards to support a cross-platform, cross-server, shared cyberspace. Living Worlds is an example of such a standard. Living Worlds builds upon standard connections of a Web browser to the Internet to provide a way of sharing 3D virtual worlds with multiple participants. Many Virtual Reality Modeling Language (VRML) based, 3D Web browsers have recently incorporated an External Authoring Interface (EAI) or Application Programming Interface (API) that allow others to write the scripts that make cyberspace come alive. Server technology is rapidly improving the shared behavior routines that allow one browser to see the effects of the actions of other cyberspace visitors. So, it seems time to start building the worlds that 3D graphical technologies will support. How effectively will we be able to communicate, educate, and entertain ourselves in cyberspace over a typical Internet connection (using a 28.8 kbs modem)? The question is a wide-open one. It is time to begin to build worlds and test out our abilities to collaborate in them.

Collaborative, shared digital worlds have been created in the past. Many have been created on dedicated networks such as the military's Distributed Interactive Simulation (DIS) network, Japanese research lab networks, or university laboratory networks. Past attempts at electronically mediated shared spaces were always created at great cost and with little opportunity to make them available to the mass public. The commercialization of the Internet and the advent of the World Wide Web have made mass public access to computer networks a possibility. The Internet is not a reliable deliverer of information and, to-date, has not been designed to be one. The attempt to port yesterday's shared digital worlds have failed. Past applications have required reliable and rapid delivery of information. Today, the technology is being built from the bottom up to work with the inherent weaknesses of Internet information delivery. The goals may be the same as

cyberspace projects of the past, but the delivery strategies are quite different.

We have an opportunity to provide cyberspace access for millions of world citizens. Some of these people will be able to truly participate in a cybersociety where they have struggled to participate in our society to date. Many physically challenged individuals are not mobile. They can't easily run out and participate in a spontaneous societal event such as a political rally or sports team celebration. Even an average world citizen can't always physically get to where the educational and entertainment opportunities are available. Instead, they struggle to find the best opportunity in their own neighborhoods. In too many neighborhoods the best opportunities are not nearly good enough. They are limited by the experience and knowledge of the neighbors who live there.

The technologies needed to realize a collaborative cyberspace are falling into place. It is time to build cyberspace as it should be built to reach out to the needs of our society. It is everyone's responsibility to build it right. I am doing my part by understanding the technology, trying it out, and providing feedback through this project paper. Living Worlds and other 3D cyberspace standards address networking, user interface, application programming interface, and avatar representation issues. I am learning concepts that will have broad, transferable teachings for me.

This project report has six chapters and this introduction. The seven chapters are organized in three main sections. In this first section, which includes Chapters 1 and 2, I provide an overall context for my work, review the history of technology supported communication and discuss what adding a third dimension to computer displays affords us.

In the second section, which includes Chapters 3 and 4, I review how the Internet assists collaboration between human beings, the current state of the art in Internet based, shared 3D virtual worlds, and various strategies for providing intuitive interfaces to computer users to help them interact in a shared virtual world. Basically, Chapters 3 and 4 document the current technologies I had available to me from which to build my own multi-user, shared, Internet-based, 3D virtual world.

Finally, in the last section which includes Chapters 5, 6, and 7, I review the goals for the shared world I built, review its design, and document its use through a simple pilot test that demonstrates its fit for purpose. Chapter 5 reviews the design decisions I made, the purpose of the world, and an explanation of the interface I provide to world visitors. Chapter 6 documents changes I made to prepare the world for a pilot test and documents the test design. Chapter 7 concludes this paper with various overall thoughts, pilot test results, and conclusions drawn from my work on this project.

2 COLLABORATION THROUGH TECHNOLOGY – A HISTORY OF CHOICES

Walk into any Fortune 500 company and talk to an employee — chances are he or she is scheduled to be involved in a meeting within the next four hours. Meetings are prevalent in corporate America because collaboration is critical to organizational success. Collaborative skills are critical for most corporate employment positions. This makes common sense in the information age. As the amount of available information increases, the individual is less able to digest it all to make the most intelligent decision when faced with a problem or opportunity. A corporation's collective knowledge is spread out among tens, hundreds or thousands of individuals. The better all desired corporate knowledge is mapped to different job positions, the better the chance the organization has of proactively optimizing its course of action. As a result, corporate knowledge is dispersed through individuals with specialties such as law, finance, engineering, human resources, and marketing. When a problem or opportunity comes to light somewhere within the organization, the course of who collaborates with whom makes a difference in the resultant action taken by that organization. There is a best possible action that is more probable of coming to light if the right people collaborate to discuss the solution. The same forces of collaboration are there when organizations collaborate with other organizations to promote an industry or develop a mutually beneficial technological standard.

Technology has always been involved in enabling better collaboration. Technologies are being developed faster than ever before. Collaboration has made faster project timelines possible. As an example, take a look at how the Virtual Reality Modeling Language (VRML) standard took shape in such a short time frame. VRML is a computer language that defines three-dimensional (3D) models. Or, as an even more current example, take a look at the Living Worlds standard being promoted to standardize how we interact with VRML worlds and provide a more consistent 3D cyberspace. The Living Worlds standard setting process is a case study in collaboration. Venture capitalists have strongly suggested a deadline by which the 3D cyberspace community shows a viable market for their technologies in order to obtain ongoing venture capital. At the Earth to Avatars conference in San Francisco in October 1996, the technologists chanted in unison, "a one billion dollar industry by the year 2000" as a goal for building multi-user worlds and communities. In order to have a shot at that goal, those involved realize they need to collaborate to decide the norms on which the technology will be built. There is no time for each individual or organization to follow their own course of action for two years and then try to market their solution. In fact, those involved with Living Worlds are trying to map the knowledge needs of Living Worlds to different organizations. Of interest is the speed and decisiveness blaxxun interactive demonstrated in changing their business direction to focus solely on multi-user servers. They had been developing a competent and compelling VRML client, Cybergate. Now, they trust Netscape Communications Corporation (NCC) and Silicon Graphics, Inc. (SGI) to develop the technology for the 3D viewer. The Living Worlds consortium includes companies and individuals with specialties across many knowledge bases. Anthropologists speak of how Living Worlds must provide an appropriate culture for participants. Physicists speak of how shared virtual worlds must follow basic laws of physics in order to attract visitors. Financiers advise what is necessary in order for people to be willing to spend money in cyberspace. All these folks are collaborating because they have something to say and because it is easy to do so. The ease in which we collaborate today has dramatically affected the frequency in which we do so. Technology has made it easier. Telephone lines have interconnected us worldwide. Computer networks have interconnected us through our computers. We speak of "off-line" meetings where information is discussed through voice mail messages, answering machine messages, and computer databases. There is no specific meeting time in which a collaborator must be present. Instead, there are deadlines by which your word need be voiced in order to have an effect on a decision.

The ability of an organization to take advantage of collaborative technologies seems to be dependent on its culture. Many of the studies on the return on investment of collaboration enabling

technologies are as inconclusive as studies on management styles. For certain organizational cultures, collaborative technologies have made a tremendous difference in organizational success. In fact, those organizations quantify returns on collaborative technology investment of over 200%. The technology seems to do best within a collaborative culture although the level of computer and language skills of each collaborator is a key determinant also. Looking out at society today, I see more information and improving computer skills. It is only natural to project those trends into the future and expect more collaboration and more off-line meetings. It makes sense to research the technologies that will enable collaboration tomorrow.

What aspects of a technology make it supportive of collaboration? The review that follows will be discussed in relation to the characteristics of collaboration friendly technologies outlined in Table 2.1. These characteristics are the characteristics I have found mentioned most often in various white papers that support the need for collaborative technologies.

Table 2.1 Characteristics of Collaboration Enabling Technologies

Efficient — allows immediate sharing of communication
Organized — allows information to be shared in a logical manner Timely — keeps information content current and appropriate
Available — can be used 100% of the time
Access — easy to get access to
Time Independence — collaborate at any time
Place Independence — collaborate anywhere
Self-Documenting — tracks the history of communication as a by-product
Emotional — captures the emotion of the collaborator
Imaginative — captures the imagination of the collaborator
Brainstorm Enabling— supports new idea generation
Iterative — allows iteration toward better ideas and understanding
Indexed — allows past communications to be easily reviewed.
Scalable — allows many to collaborate simultaneously
Precision — allows for a precise representation of facts
Immersive — captures the full attention of the senses

Some of these characteristics are largely subjective. For example, rating a technology in terms of its ability to support emotional communications is not an exact science. When I do rate a technology based on a subjective characteristic, much of the rating is my solely my opinion from using the technology on multiple occasions.

The remainder of this chapter looks at the history of technologies that have helped foster collaboration between human beings. We should not ignore the human aspects of collaboration. Before I review the technologies that help us collaborate, I must mention the collaborator. Collaboration is a skill just as negotiation, arbitration, communication, and persuasion are skills. Humans can be taught to enhance these skills. Child development researchers have found that we develop collaborative skills early on in life, yet learn to discredit some of those skills to survive in a competitive world. To prepare for work at a collaborative organization, our youth require an education that is consistent in providing opportunity to collaborate and that rewards collaborative behavior when demonstrated appropriately. There is a balance that can be developed such that the individual is both personally enterprising, yet organizationally collaborative. So, the logic follows that a collaborative individual will do well in promoting a collaborative organization that will take advantage of collaboration-enabling technologies.

I will include collaborative skill building technologies in my collaboration enabling technologies review. Much successful game playing requires collaborative skills (Parker Brothers' board game Risk being a well known example) and therefore I will include technologies such as Multi User Dimensions (MUDs) as part of my review. They enable people to interact in a shared experience that can be for entertainment, education, or group communication.

I will review eleven technologies that enable collaboration. The timeline in Figure 2.1 depicts them all in terms of the year they were developed. Each technology added a new capability to technology-assisted collaboration. Paper, telephone, Email, MUDs, video games, DIS, Lotus Notes, Inter Relay Chat, Greenspace, and Avatar Based Multi-user worlds all have had a significant impact on making technology more useful for collaboration. I will also review video conferencing that has been around since the day of the first video telephone which failed miserably.

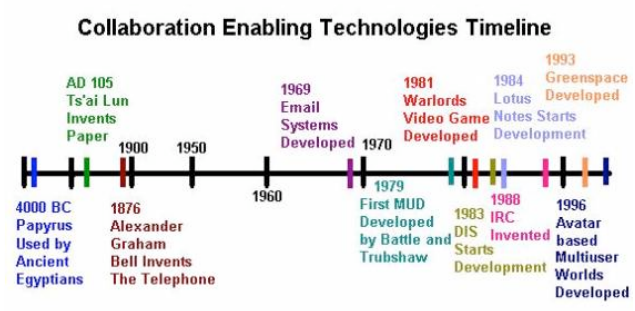

Figure 2.1 Collaboration Enabling Technologies Timeline

2.1 Paper

According to the Mead Corporation, ancient Egyptians invented the first substance that crudely resembled paper around 4000 B.C. *Papyrus*, as they called it, was a woven mat of reeds, pounded together into a hard, thin sheet. Afterward, the Ancient Greeks used another paper-like substance made from animal skins. Paper as we know it today was invented by Ts'ai Lun, a Chinese court official, in A.D. 105. [1] Paper was important to the evolution of collaboration as it enabled communication among multiple people who no longer had to be in the same place at the same time in order to communicate. Paper was the first significant improvement to the collaborative process since language had evolved and was used for effective communications. Paper facilitated the evolution of the written word and improved dramatically our knowledge of history by creating the medium for its archival. Paper today is still a significant part of many collaborative processes. It is inexpensive, durable, expendable, recyclable, and continues to evolve. Still, the most sophisticated collaborative teams use paper to review information on the bus, on the beach, and in reports created for a mass audience. Individuals without computer skills continue to be given the option of reviewing paper output and providing paper input to a collaborative process.

Evaluation of paper. Paper falls short as an efficient or organized collaboration technology. Multiple copies of the same basic facts are produced and rearranged in order to provide a different sequencing. Written text is sequential and requires additional indices in order to be randomly accessed. Paper indices provide an appropriate page number, but a collaborator still has to physically turn to the referenced page. Paper is not necessarily timely. As improved or updated copies of a paper-based information source are created, the older paper documents still physically exist and become sources of inferior information. Paper is readily available and widely accessible. Paper provides time independence to collaboration, but is place dependent as paper consists of physical atoms that must exist within the collaborator's eyesight. The author can use paper anywhere, but the reader is clearly restricted in obtaining access. Paper has been used often solely for its self-documenting ability. A blank sheet of paper provides an outlet for emotion, imagination, and brainstorming, yet relies on the written skills of the collaborator. Paper is a poor medium for iterative tasks and requires significant work to be indexed. Paper is not very scalable although a paper copier can quickly produce copies of results of the collaboration process. Paper can be used to create precise communications, but it is a 2D medium that has trouble representing three dimensions. Reading words on paper and writing on paper are not especially immersive experiences, yet the act of reading and writing does seem to occupy the mind's attention such that the other senses are ignored to some extent.

2.2 Telephone

Alexander Graham Bells successfully demonstrated his telephone invention on March 10, 1876. The telephone advanced the spoken word as paper had advanced the written word. At the time, through a postal service, the written word could be shared among collaborators who never had to meet. The telephone advanced that unique luxury to the spoken word and added an additional benefit of a more instantaneous collaboration.

Evaluation of the telephone. The telephone is a more efficient technology than paper with respect to immediacy. Over long distances, information can be shared as fast as electricity can travel the distance by wire. Yet, the telephone does no more than paper to organize information as information is provided sequentially. The timeliness of information shared by telephone is only as current as the last conversation. Telephone availability is near 100% in most first world countries, yet still growing in third world countries. Telephone access has increasingly improved since 1876 although accessibility started especially slow in its early days. Accessibility is taking another leap with the advent of the cellular telephone. Unlike paper, the telephone requires some time dependence although voice mail has eliminated much of that requirement. The place independence of the telephone is related to its accessibility and the increasing development of the number of cellular phone "cells" that carry the communications is changing the place dependence scale. For a live conversation, the initiator can be anywhere a telephone is available, yet the recipient must be in a place that is aware of the ring of the initiator. In many cases, this is a significant shortcoming.

The telephone is not naturally self-documenting. Conversations can be recorded, but even then need to be reviewed sequentially. It is more difficult to review recorded voice documentation than a paper-based document. The telephone captures the emotion and imagination of the collaborator, yet only through verbal communication. Voice inflection can make emotion more obvious than the paper-based written word. The telephone has no specific advantage for iterative communications and is a poorly indexed technology. Teleconferencing has improved the scalability of the telephone, but its scalability still falls short of ideal. The telephone lacks the richness to efficiently communicate precision and is especially weak on visual images. The telephone is hearing immersive, but ignores the other senses. Yet, it complements the partial visual immersion of paper well as the number of occurrences of phone calls made to discuss a paper document suggests.

2.3 Electronic Mail

The idea of email arose in the early multi-user systems and research laboratories of the late 1960s. The United States Department of Defense's Advanced Research Projects Agency had developed ARPAnet, (which in 1969 became the Internet [3]), and it was expanding quickly. In the business sector, email easily suited host-based systems in which large numbers of users were connected by terminals. Proprietary host email systems such as IBM's Professional Office System (PROFS) or DIStributed Office Support System (DISOSS) and Digital Equipment Corporation's (DEC) All-In-1 or VMSmail were popular tools of the time.[2] Email is a widely used communications and collaboration tool because it enables people or mail-enabled applications to exchange multimedia information, workflow, and electronic data interchange transactions.

Evaluation of electronic mail. Overall, electronic mail combines many of the benefits of paper and the telephone. First and foremost, email is a more efficient technology than paper. Email can be organized more easily than paper or telephone correspondence as it is in an electronic format that the computer can use to organize. Still, email does not naturally keep information any timelier than paper or the telephone. Computer based services can communicate information changes by maintaining mailing lists and each recipient can permit the latest incoming message to replace the last transmission. Paper based subscriptions provide the same service but require the recipient to replace the last transmission manually.

Availability of email has been problematic to this point, but there is no real reason it shouldn't be as available as the telephone. Access is currently more difficult than the telephone, but again there is no technical reason for its inferiority. Email is exceptionally time independent and gets closer to complete place independence daily. Email is as emotional and imaginative as paper as they both rely on the written word, yet typically, email is less brainstorming enabling than paper when anonymous messaging is not available. Still, email lacks the voice emotion capabilities of the telephone. Email can be set up to be more iterative and indexing than paper or the telephone. Again, logical computer processes can be used to maintain iterations and a randomly accessible index over time. Email is very scalable compared to the telephone or paper as it takes advantage of the client/server benefits of computer networking technologies. Email is no more precise than paper or the telephone and no more immersive than paper.

2.4 Chat

Jarkko Oikarinen wrote the original Inter-Relay Chat (IRC) program at the University of Oulu, Finland, in 1988 [4]. He designed IRC as a client/server program. IRC differs significantly from previous synchronous communication programs. Fundamental to IRC is the concept of a channel. Original chat

programs had no need of channels since only two people could communicate at one time, typing directly to each other's screen. Other chat systems have been developed with similar features to IRC. Basically, chat is the text based equivalent of the telephone. Chat technology is similar to a teleconference in that, unlike with paper and email, the telephone and chat allow for more interactive collaborations as any collaborator can start communicating information at any time during their use of the technology.

Evaluation of chat. Chat is very efficient as it passes every written thought to each subscriber to the channel immediately. Chat is extremely unorganized and often described as chaos. The collaborators that use chat can set up some protocol ahead of time, but chat does effectively nothing to enforce it. Chat's timeliness rating is similar to the telephone's. Availability and access are similar to email. Chat is more time dependent than email because it is real-time, yet no more place dependent. Chat tends to be more emotional than email not through its form, but because it is so immediate and emotion is raw, often overcoming typical inhibitions of the communicator. Chat's biggest benefit over other technologies is in its brainstorming potential because anonymity is assured and new idea discussion can be quick and rapid.

Chat could be as iterative and indexed as email, but the supporting computer processes have not been developed nor applied to chat to date. Chat is as scalable as email because of its client/server nature. Yet the more concurrent users, the more chaos creeps in to the collaboration process. Chat is no more precise than email, yet tends to be more immersive than email because so much is happening so quickly and that intensity of communications requires more attention.

2.5 Video Conferencing

Video conferencing is a collaborative technology where multiple cameras and microphones provide simultaneous voice and images of collaborators such that the collaborators can see the images of all other cameras except the one focused on them. All collaborators hear all voice transmissions. Video conferencing is used often used to replace travel when collaborators feel the need to see one another while collaborating.

Evaluation of video conferencing. Video conferencing is similar to chat in many respects, but tends to be more organized as collaborators gain access to more communication feedback from other collaborators. Video conferencing provides voice and gesture feedback as collaborators can see and hear one another. The enhanced feedback comes at tremendous expense as simultaneous voice and video require significant bandwidth while chat bandwidth requirements are trivial. Video conferencing tends to limit the intensity of the emotional response from each collaborator and severely limits the brainstorming anonymity of chat. Video conferencing is also more difficult to scale to many different locations. Video conferencing can provide a more precise representation of 3D information as the camera can move around within a 3D space. Video conferencing provides partial immersion of more senses than chat, but the experience seems less immersive than an intense chat session. Video conferencing is currently more place dependent than chat but, with infinite bandwidth, need not be so. Similar to email or chat, the computer can be used to organize and index collaborative information. Although algorithmically more difficult to index video than text, the Motion Picture Experts Group's latest video standard being researched (MPEG4) is attempting to index video by the significant events that appear on camera.

2.6 Multi User Dimensions (MUDs) and Object Oriented MUDS (MOOs)

MUD1 was the first proper, workable multi-user adventure game using text-based communications over a computer terminal. Roy Trubshaw and Richard Bartle at Essex University in England wrote MUD1 on a DECsystem-10 mainframe. Trubshaw began in autumn 1979, and Bartle took over in summer 1980. Initially, the game was playable only by students at the university and guests using the university's system. After a year or so, however, external players began to direct-dial from home using modems, and the game's popularity grew. [6] MUDS and MOOS are still very popular for social and game-based collaboration.

Evaluation of MUDS. MUDS and MOOS can be considered chat with a context. Usually, a MUD collaborator has a sense of presence in a 3D environment that helps focus communications on his or her surroundings. The MUD is housed on one or more computers that contain the details of the world as well as act as the chat server. Because of the context provided by the world database, communications tend to be more organized than with chat. The communication is just as efficient as the telephone. Since the computer makes changes to the database over time, MUD based information is more timely than the telephone. MUDS tend to be less available or accessible than email or the telephone because concurrent use is usually restricted in order to keep up the quality of service.

MUDs are as time and place independent as chat and can be self-documenting if some logging service is provided on the world computer. MUD collaboration can be as emotional as chat. In fact, quite an elaborate subculture has arisen in the MUD community such that text based norms have been promulgated that creatively attempt to make up for the lack of voice and gesture feedback of keyboard-based transmissions.[7] MUDs are an extremely imaginative collaboration technology as collaborators are often allowed to add rich 3D based text additions to the world database. Collaborators must use their imagination to see the world they are investigating. Brainstorming is limited by the fact that the MUD usually already has its context determined by its choreographer, but a collaborator is provided the benefit of anonymity.

MUD scalability is not limited by technical considerations, but by the sense of community. A larger, faster computer or bank of computers can always be used to house the MUD, yet more collaborators tends to disturb the sense of peace of the MUD experience. MUDs provide a collaborator a means for specifying a third dimension, but only as precisely as words can detail it. MUD participation is extremely immersive as the mind is occupied by creating a picture of the world from a text-based description. No aural or haptic immersion is involved.

2.7 Networked Groupware

Version 1.0 of Lotus Notes was developed from 1984 to 1989 through the design and programming efforts of Ray Ozzie, Len Kawell, Tim Halvorsen, and Steve Beckhardt, the first three of which had developed a strong vision of groupware from having worked with the Plato system at the University of Illinois in the mid-1970s. At the Computer-Based Education Research Lab there, an electronic newsgroup-like computer program called gnotes, run by users in remote places sitting at a Plato terminal, allowed users to share group messages. Although Plato terminals

were attached to a mainframe, the environment had the feel of today's PC networks. [6]

Groupware such as Lotus Notes organizes collaborators' off-line discussions, creating discussion threads, multiple indexing, and time stamping. Groupware databases are becoming more graphical as images are easily inserted into the body of the text. There are common sense arguments supporting off-line collaboration over face to face meetings such as the following:

- Participation independent of location to save travel time and conflicts
- More time to think about new information before responding thereby better response quality
- Consideration of most relevant information first in order to think top-down
- Ability to skip details of issues not relevant to a participant in order to save time and resources
- Participation when feeling more participative and energetic
- Revision of thoughts before presenting them in order to avoid miscommunication

Evaluation of groupware. Groupware is as efficient as email, yet significantly more organized as a typical groupware database contains more than just a few dynamic indices to the database's documents. These views as they are called are easily created by available sorts and filter and provide a sophisticated search capability. Timeliness is improved over other technologies as only one copy of a document exists in the database at any time and is updated to remain current. Groupware, like email, continues to become more available and accessible. Both can be considered as 100%. Groupware is both time and place independent and does a great job of self-documenting the collaborative process. In fact, a selling point of groupware is its benefit of tracking historical collaboration for later use in confirming details or revisiting a decision point that was influential in success or failure. The documentation process then helps others learn how to collaborate.

Groupware is similar to email in its emotional, imaginative, and brainstorm enabling ability. Because of its organization ability, iterative collaboration is better served by groupware than all other collaboration enabling technologies. Groupware is scalable through its client/server architecture. Groupware allows for a precise representation of facts through text and images, but is not yet considered a 3D medium. Groupware is as immersive as paper-based information sources.

2.8 Networked Video Games

Video games were born when the first computer display was used to represent information in a graphical format and a user interactively moved the image with a goal in mind. Video game evolution has been constant since that point. Video games evolved rapidly in the golden years of video games at the beginning of the 1980s. In 1980, the video game Defender became the first video game with a virtual world where activity was happening outside of the physical view of the player. Also in 1980, Battlezone became the first truly interactive 3D environment used in a video game and Bezerk added the first spoken vocabulary of 30 words. Finally, in 1981, the video game Warlords became the first collaborative game where cooperation with other players actually helped a player gain a higher score [11].

Since then, video games have been networked over long distances and the graphical displays have continued to become more impressive. Video game technology is very efficient, organized, and timely as the computer controls all three aspects. Video games are as available and accessible as other network computing technologies, yet often require higher bandwidth resources than email or chat. Since the game typically changes often and instantaneously, there is a high time dependence although the place independence improves at the rate of new network bandwidth rollout. Video games are not usually self-documenting, but can become so with added overhead. Video game users can demonstrate emotion through the characters they represent. Their playing piece typically demonstrates human-like, non-verbal communication. Video games can capture the imagination of the player, yet with such exact graphical output, usually are very literal. Imagination is more associated with immersion in the game-playing environment where the player imagines being the playing piece in the scene. Video games are usually not brainstorm enabling. Video games can be built to be iterative toward better understanding and can be indexed along the way such that a saved game condition can be loaded and revisited. Video games can be scalable through a client/server architecture. A precise 3D representation of data can be represented using video game technology and video games can become an immersive experience of all the senses.

Video games will continue to push collaborative technologies through their money making potential. Today's networked video games like Doom and Quake make millions for their creators as children and adults are willing to pay significantly for their entertainment.

2.9 Distributed Interactive Simulation (DIS)

According to the DIS Steering Committee:

> "The primary mission of DIS is to define an infrastructure for linking simulations of various types at multiple locations to create realistic, complex, virtual "worlds" for the simulation of highly interactive activities. This infrastructure brings together systems built for separate purposes, technologies from different eras, products from various vendors, and platforms from various services and permits them to interoperate. DIS exercises are intended to support a mixture of virtual entities (human-in-the-loop simulators), live entities (operational platforms and test and evaluation systems), and constructive entities (war games and other automated simulations) [8]."

In many regards, DIS can be considered a video game technology. DIS is considered a significant standard for potentially networking hundreds of thousands of users together in a virtual world based simulation. DIS architecture is different than a typical network game server. The first successful DIS prototype started development in 1983 and was unveiled in 1989.

Evaluation of DIS. Compared to video games, DIS is different on the emotional scale as a collaborator never controls the human

form. Non-verbal communication is limited. Collaboration is through voice discussions of strategic choices of action. DIS is more scalable by nature of its design. Since the DIS standard is evolving toward improvement on all the collaboration scales, the interest lies mainly in what it will become.

2.10 Greenspace

Greenspace is a 3D virtual environment platform developed by the Human Interface Technology Laboratory at The University of Washington. Greenspace technology is based on multicast, collaborative, object-oriented classes driving 3D stereo visual display, spatial sound, speech recognition and synthesis, position tracking, touch and gesture input with force feedback. [10] Although all design goals have yet to be met, the technology encompasses the ultimate in fully immersed virtual collaboration. The key to Greenspace is the position tracking of the body of a collaborator in order to represent the collaborator's actions in the virtual world. The first phase of Greenspace was demonstrated in November of 1993 while the second phase was demonstrated in 1995.

Evaluation of Greenspace. Since the collaborator actually becomes a virtual human, non-verbal communication is enhanced over the typical video game. Multicasting permits Greenspace to be more scalable than today's video games. All other aspects of Greenspace rank similarly to the video game collaboration evaluation.

2.11 Avatar based Multi-user Worlds

True 3D multi-user worlds based on VRML became publicly available in 1996 from many different sources including blaxxun interactive, Intel Corporation, and Sony Corporation. Avatar based multi-user worlds are a combination of chat and the Greenspace ideal. Built with lower bandwidth requirements, multi-user world technology relies on many of the aspects of chat and MUDs that have made them successful as collaborative tools. Multi-user worlds allow for text based communications and unique identification of the user in the world through an avatar. Behaviors need not be human, yet the avatars provide some form of non-verbal communication that can be interpreted by a human. Like MUDs, they can be quite social in nature. Voice based chat is rapidly being incorporated into multi-user virtual worlds.

Evaluation of multi-user worlds. Multi-user worlds are as efficient as chat, and are as organized and timely, but with a different purpose. Multi-user worlds are organized around a very visible three-dimensional information space. Any changes to the three dimensional state of the shared world are updated in a timely manner. Availability and accessibility are similar to MUDs as are the time and place independence ratings. Multi-user worlds do not have any explicit self-documenting feature, but the self-documenting ability can be built within the computer process with overhead. The inclusion of an avatar allows the collaboration to be more emotional than with MUD technology, but significant work is still needed to make useful avatar behaviors a reality. Multi-user worlds are more literal than MUDs that can be interpreted as being less imaginative. Yet, for communicating truly unique 3D images, multi-user worlds can capture the imagination of the collaborator. The brainstorming ability of multi-user worlds is dependent on the tools available within the world.

Multi-user worlds can provide iterative collaboration of 3D design in pieces, or all of the world can be replaced with a new geometry to communicate a better idea or understanding. Since multi-user worlds are Web-enabled, objects can be brought in from any Web server to aid communication. Multi-user worlds are not as immersive as Greenspace, but can become so as the technology develops.

2.12 Conclusion

Technology has been enabling collaboration for centuries now. As time passes, the best features in one collaboration enabling technology are appearing in the others. For example, the benefit of place independence provided by email is being extended to telephones through cellular technologies. Or, as another example, avatars are being added to networked computer programs to enable non-verbal communications in a virtual world. Some of the features of collaborative tools are actually in competition with each other. For example, MUD users mention how they appreciate the personal interpretation they are afforded by being provided a text only communication medium. Their interpretation is personal because the pictures they create are solely inside of their head. As we move to Greenspace environments, the picture is very literal. A literal picture has some powerful benefits, but the trade-off is always a loss of personal interpretation, something our society often defends as personal freedom.

In collaborative technologies of today, convergence is the key. A collaborative environment taking advantage of a multi-media enabled computer can pick and choose the best aspects of each technology reviewed in this historical perspective. As computer bandwidth and processing speeds improve, collaborators will expect all of the best collaboration enabling aspects of technology in the same collaboration tool.data model and data acquisition.

3 3D COLLABORATION ON THE INTERNET

3D, multi-user collaboration on the Internet is currently enabled through five component technologies: 3D world modeling technology, communications technology, Web browser technology, server technology, and client/server connection technology. Each plays its part in providing the multi-user experience to each collaborator. The world model defines which visual objects are part of the collaboration and defines each object's geometry, appearance, location, orientation, and scale. Communication technology defines how collaborators communicate: via text, voice, or email while participating in a multi-user world. Browser technology defines how each collaborator interacts with the world model and how the other communications technologies are integrated with the world model interaction. Server technology coordinates the collaboration to make the collaboration process sensible to each collaborator. Client/server communications technology enables the browser to communicate with the server(s). This chapter provides a component-by-component review of the current state of 3D world based collaboration over the Internet.

3.1 The World Model

An author typically uses a 3D modeling tool to create the world model. The model is created visually using a computer program with a special graphical user interface that makes it easy to point and click on points in 3D space to insert objects and define their geometry appearance, location, orientation, and scale. VRML models can be created quite effectively using the 3D Studio Max modeling package from Kinetix or the Alias modeling package

from Aliaslwavefront. Both these packages use four simultaneous views to model objects: top view, front view, side view, and perspective view. Figure 3.1 shows an example of a world-modeling package. The state of the art today in virtual world modeling allows world model developers to attach object behaviors to the objects being modeled. Modeling tools are used to create each collaborator's personal representation, called an avatar, which enables non-verbal communications within the multi-user world.

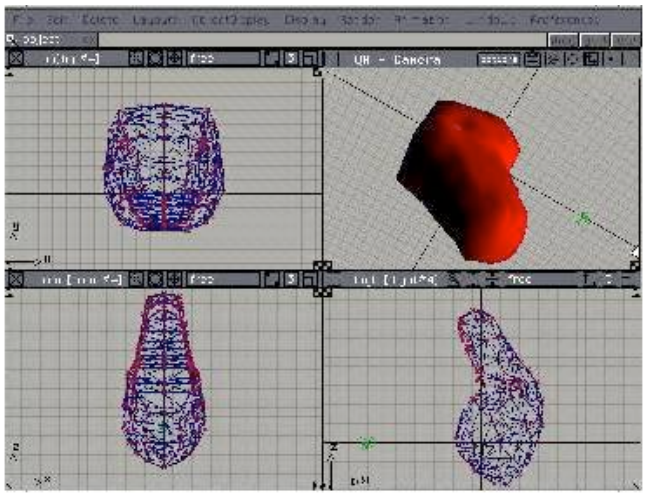

Figure 3.1 The Alias' 3D Modeling Interface

Once the model is created, a world model must be delivered to each participant in order that each collaborator has access to the same images on his or her monitor while collaborating. The Internet has improved world model delivery significantly as a collaborator today only needs access to the Internet in order to gain access to the latest world model. VRML is a standard that defines a world model and stores it in a simple ASCII or UTF8 text-based file[12]. VRML files can be delivered using http: Web server delivery strategies. Because of the World Wide Web craze of the early 1990s, Web servers have become optimized for delivery of HyperText Markup Language (HTML) documents over the Internet. HTML documents are simple text-based documents that define how a Web page is viewed by a Web user in a Web browser [13]. HTML documents can obtain components of a Web page from other Web servers, allowing efficient use and reuse of text and graphical information. Within the VRML standard, object geometry, appearance, location, orientation, and scale are defined for each object in the virtual world. VRML then puts the objects together to make a 3D scene. VRML uses the same Web servers that HTML documents have used so successfully. No additional enhancements are needed to deliver 3D models anywhere on the Web. And, VRML has the same facility for obtaining world components from anywhere on the Web and including them in the current 3D world that HTML has with text and 2D graphics.

VRML objects can also be stored on and loaded from a local hard drive or CD-ROM drive. Other shared world model providers use other modeling technologies and deliver the content locally before a collaborator begins to use the technology. Video game technologies have traditionally required local storage of the game world before a player connects to a shared experience. The technology is downloaded using the Internet or is purchased in a physical store in diskette or CD-ROM format. VRML has been designed specifically for Web server delivery that is more efficient for rapidly changing world content. The collaborator always accesses the latest world model because she obtains it from a Web server that only houses the latest world. Web server delivery continues to evolve and it seems sure that other technologies will provide on-line, server-based 3D world model delivery similar to VRML.

The world model author develops the model while considering its final file size and its complexity. The file size is considered relative to expected storage capacity and available Random Access Memory (RAM) of a user's computer as well as estimated download times for Internet accessed files. The world complexity is considered relative to the 3D multi-user client's capability to manage the complexity within a reasonable frame loop explained in the Browser section of this chapter..

3.2 Communications

As seen in Chapter 2, collaborators have been using different technologies to enable communications during the collaborative process. These communications technologies are all becoming available in 3D shared multi-user worlds. As to their coordination with the 3D world itself, many voice, text, or email implementations are best considered as separate, parallel computing processes. Telephony, voice-enabled or text-based chat, and email can be used in separate windows or frames in order for collaborators to discuss the 3D world they are sharing. Also, collaborators can choose to use separate tools and share information through a separate communications server that does not rely on any coordination with the shared world.

There is some benefit to integrating the communications within the 3D world. If an avatar modeled as a 3D object represents a user, others are aware of his location in 3D space. If the chat environment is aware of the user's location, it can take that information into account to modify communication messages. An obvious example of this is the voice-enabled chat environment provided in On-Live! Technologies Inc.'s virtual world technology. In that environment, the volume of a collaborator's voice is dependent on his or her distance from another collaborator. This set-up is a natural interface that provides easy one-on-one communications. Two collaborators need only find a location away from other users and they will only hear each other. If six users stand in a circle, they all hear each other at the same time with similar volume.

Even text-based communications can take advantage of the 3D locations of the collaborators. Distance from others can dictate which chat channels a user is privy to. A new visitor on the scene can quickly determine who is available for a chat session based on their location in 3D space. For example, blaxxun interactive's Passport client provides a beam feature that allows one collaborator to quickly move to a location in the world that is directly in front of another collaborator identified by name. His or her avatar is also automatically oriented to face toward the other collaborator. At that point, it is clear that a one-on-one chat session is being pursued and the collaborators can negotiate to establish it. For integrated communications scalability, the communication process can run on a separate server or as another process on the same server.

The collaborator communications decision is a difficult one for a 3D multi-user developer. The more communication bits coming

over the Internet to a collaborator, the more bandwidth he or she needs to be able to manage them in real time. To develop for a 14.4 or 28.8 kbs modem connection, trade-off decisions have to be made. For example, On-Live! Technologies Inc.'s Traveler viewer does not provide a body for a collaborator's avatar since the animation of the mouth bits and voice bits occupy a significant portion of the bit budget supported on a slow Internet connection with limited bandwidth.

Figure 4.1 The Passport Viewer (courtesy of blaxxun interactive)

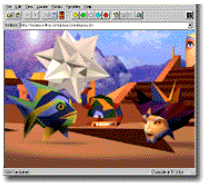

Figure 4.2 The Traveler Viewer (courtesy of OnLive! Technologies Inc.)

Figure 4.3 The CyberPassage Viewer (courtesy of Sony Corporation)

Figure 4.4 The OZ Virtual Viewer (courtesy of OZ Interactive Inc.)

3.3 The Browser

Most clients used by 3D world collaborators use two separate applications. A Web browser originally designed for HTML document presentation provides General Web navigation. 3D model specific navigation is provided by another application that communicates with the Web browser through a defined specification; Netscape Communication Corporation's plug-in Application Programming Interface (API) being the most popular. Almost all popular VRML viewers, for example, connect to the popular Netscape Communications Corporation's Navigator and Microsoft Corporation's Internet Explorer Web browsers through the plug-in technology originally provided by Netscape Communications Corporation. The Web browser communicates with Web servers to request and send information on the VRML viewer's behalf. The VRML viewer then uses the data received to create the 3D world seen by the user. Yet, stand-alone 3D VRML viewers do exist and provide refreshing, creative clients. OZ Interactive Inc.'s OZ Virtual browser is an example of a VRML browser that handles basic Web navigation without the help of another application.

The client is responsible for parsing the world model as it is delivered from a Web server, determining a beginning viewpoint, rendering the scene based on that viewpoint, and then maintaining changes that occur based on the user's interaction with the scene or server-based messages sent to the client. Incoming bits represent incoming text or voice communications, avatar location changes, behaviors in the shared world initiated by any collaborator's actions or timers embedded in the world. The local collaborator's viewpoint is managed locally within his or her client. Outgoing bits include changes to the local avatar position, behaviors activated by the local collaborator, and text or voice messages sent for communication with other collaborators.

Most of the obvious differences between browsers are a result of different choices in the look and feel of the user interface. This chapter will take a look at the basic capabilities of most 3D multi-user enabled browsers. blaxxun interactive's Passport [14] (Figure 4.1), On-Live! Technologies Inc.'s Traveler [15] (Figure 4.2), Sony Corporation's CyberPassage [16] (Figure 4.3), and OZ Interactive Inc.'s OZ Virtual [17] (Figure 4.4) are all popular 3D multi-user world browsers. New versions appear approximately every month and, as a result, make any description of their capabilities outdated soon after putting the words on paper. The following comments are as of February 1997.

All four browsers handle voice and/or text chat simultaneously while providing a shared world model to each connected user. Each provides an avatar that can be seen by others as a representation of each user. All are working to incrementally incorporate the VRML 2 standard. The VRML 2 standard provides object behaviors such as change in location, orientation and scale, change in color and lighting and appearance and disappearance. The user has control of a navigation mode such as walk, fly, or examine mode, the ability to turn on or off a default headlight attached to his or her virtual head, the ability to bookmark an exact location and orientation in a world, and the ability to enforce collision with other objects or disable it. The user moves around in the world using the arrow keys on the keyboard or by way of mouse movement within the world itself or relative to a control panel provided within the interface.

These clients provide each collaborator the ability to choose their avatar from a avatar collection. Some allow each avatar child object (such as hat, shirt, shoes, etc.) to be changed or colored separately, and some allow the user to provide their own avatar following some guidelines and VRML 1 or 2 design.

The client is carefully engineered such that a certain minimum frame rate is maintained if the minimum recommended CPU, RAM, video board, and Internet connectivity technologies are used by a collaborator. The frame rate, or number of times per second the world is re-rendered to the screen, is a critical success factor for most users. Internally, the client makes constant tradeoffs between available changes provided by all incoming bits, user movements, and mouse clicks. The higher the frame rate, the better the experience. In an ideal situation, all state changes are easily handled by the client frame loop with time left over. Then, the client can increase the frame rate above the minimum rate used internally, perform some other function, or just wait for the next frame loop to begin. When all state changes can't be handled within the minimum frame rate loop, the client can throw out some of the changes or queue them for later

processing. Each client developer creatively programs these trade-off decisions which then become more important as the world complexity increases, number of collaborators increase, and active behaviors increase.

3.4 The Server

In its most basic form, the server simply connects users together in order to send changes from one collaborator's world model to the other collaborator's world models. A collaborator connects to a server over the Internet, is delivered a current model of the world from the server or another collaborator, and then interacts with the world by sending updates of his or her actions and receiving updates from others' actions.

In its most complicated form, the server can be transforming the world itself and communicating its changes along with changes from other collaborators. The server can contain logic that monitors each collaborator's actions and regulates it in any way. For example, blaxxun interactive's CyberHub server can mute a collaborator on behalf of another collaborator's request. Today, most of the server capabilities are tied to a specific client technology. Connecting to a server without a specific client technology makes little sense.

It is possible to architect a 3D shared world without a server if each client knows how to communicate with all other peers. Such server-less multi-user worlds, called distributed worlds, usually are enabled with a broadcast or multicast communications environment. Greenspace is an example of such a multi-user world delivery environment [18]. Greenspace clients are connected over a dedicated network such that changes within one collaborator's world are communicated only to certain peers. Since the Internet IP multicast communications protocol is still in its infancy and Internet broadcasting is a tremendous waste of messaging, centralized servers have been used extensively to both deliver 3D virtual worlds and maintain communications between users for Internet implementations.

3.5 Client/Server Communications

In a client/server architecture, each function point is strategically placed at the server or on the client based on the ability and capacity of each technology. Technologists take advantage of client/server architecture to split the development effort among mutually-exclusive programming efforts. Client specialists work to make the client more user friendly and capable. Server specialists work to make the server more secure, fast, and capable. As long as the client/server communications piece is defined ahead of time, each can work on independent timelines because the latest server will work with the latest client and vice versa. Client/server development strategies are seen everywhere Internet enabled multi-user worlds are being created.

Sony Corporation, Silicon Graphics, Inc., Chaco, Intel Corporation, blaxxun interactive, and Netscape Communications Corporation all focus on either a multi-user client or server or both. The clients continue to improve to add new features and become more efficient. The servers continue to become more secure, fast and functional. And, often, the clients are upgraded monthly while server releases appear every six months. The most troubling constant is the latency brought on by an Internet connection that requires developers to respect a certain time lag between a server sending bits and a client receiving them.

The design of the client/server communications piece is critical to the technology's success. The VRML community continues to extend VRML to handle client/server communications. Yet, VRML viewer developers are creating client application programming interfaces (APIs) that let the browser communicate with a server written in any programming language. Both approaches show much promise for dynamic multi-user virtual world development. The next two sections contrast the two approaches to client/server communications.

3.6 Extending the VRML standard

The Living Worlds standard extends the VRML 2 standard through the PROTO and EXTERNPROTO node keywords in order to add multi-user capabilities to a VRML 2 scene [20]. The VRML 2 PROTO node allows an author to encapsulate all characteristics and behaviors of a VRML 2 object and make that prototype available to all other VRML scenes by way of the Web. Another author can use the same PROTO node in his or her scene by referring to it in an EXTERNPROTO node. The EXTERNPROTO node includes a field that points to the original PROTO node on the Web. The Living Worlds standard prototypes new nodes necessary for multi-user communication and object shared behaviors and makes them a standard interface for multi-user server developers to develop server software that is able to communicate with the multi-user worlds loaded in each visitor's Web browser.

Within the Living Worlds architecture, VRML 2 authors can use a prototype node called Zone to include VRML objects in a multi-user area. The Zone node is a grouping node that tells the world server where multi-user behavior is to be enabled. VRML objects can be added and removed from the Zone nodes on the fly using ROUTE statements that are an integral part of the VRML 2 standard. The most interesting nodes to add to the Zone group are SharedObject nodes because a SharedObject demonstrates its behaviors to everyone within view of the object. A good example of a SharedObject is a pair of dice that can be rolled in a multi-user game. Those dice would be added to a Zone that contained all the game pieces, game board, and game table. The game table would be a simple VRML object with no shared behaviors. Any object that is a child of a SharedObject node can at best only demonstrate behaviors to the local collaborator that initiates them unless a local timer is provided to each collaborator during initial world acquisition. These local timers can enact behaviors in each collaborator's world without the server.

A SharedObject can demonstrate the standard behaviors defined in VRML 2. If a multi-user server developer wants to create new technologies that can be enabled in a Zone, the Living Worlds standard provides a PrivateZone node that can contain a MuTechZone node and many PrivateSharedObjects nodes, each which can contain a MuTechSharedObject node. The word MuTech is short for a multi-user technician. These four nodes are all made into prototypes using VRML 2 syntax and as such only require a standard VRML 2 .wrl file to enable multi-user world interaction on each client. Unless, of course, the multi-user technician requires additional executable files to reside at the client in order to participate in their unique technology. Those files are downloaded once over the Internet from the server provider and then accessed by all subsequent VRML 2 scenes.

The Living Worlds standard-setters realize that some of the features of multi-user technology are better provided by the browser. Still, until the browser developers make those features available, an alternative way of providing rich multi-user experiences is provided through the Living Worlds standard. A Living Worlds-like methodology could be provided to any Internet multi-user world syntax where the world model itself includes the logic to initiate the server routines referenced. Such a

methodology requires the server to be on the more sophisticated end of possible server types (versus a simple message pass-through server) until the world generation syntax matures. VRML is just one standard world syntax that is getting the most publicity today.

3.7 Client based APIs

An alternative approach to providing multi-user capabilities on the Internet is to open up a world to other processes that call basic functions available within the client [21]. Considering this approach, the VRML 2 standard becomes important only for its ability to define the world objects' appearance, geometry, location, orientation, and scale and the ability to change those parameters on the fly. Outside programs can determine the changes to be made and then call the world functions available in the world viewer client that then make the changes, including adding new objects and removing existing objects.

The processes that request changes can be written in any programming language and reside on the client or a server to which the client is connected. This approach allows much flexibility for the software developer. The VRML 2 external interfaces usually work as follows:

1. The programmer creates variables in his or her program(s) that point to nodes in the VRML 2 scene graph.

2. The programmer uses the variables in her program(s) to change variable states based on programming logic or event processing (available events are triggered by timers, mouse clicks, and proximity to objects).

3. Periodically, the programmer requests the VRML 2 scene graph to update its state (and render the scene) based on the changes taking place within his or her program(s).

In this case, each event is communicated to a server and sent to all connected clients that can see or are otherwise affected by the event processing. The server is written in an appropriate language and receives event notifications from a client when a client validly changes its copy of the world. With this architecture, the server can be simple or complex as behavior generating processes can be placed at the server or the client. The server need not even keep its own version of the world if it is designed using a simple message pass-through architecture.

3.8 Considerations

The benefit of having a standard such as VRML 2 is that content authors can create content which is viewable by an audience using all kinds of different browsers. If the browser is standard compliant and the author's work is standard compliant, the content should work on the browser without ever being specifically tested by the author on that browser. Many authors and browser developers have subscribed to the VRML 2 standard in order to reap these expected benefits.

The Living Worlds standard is an extension of the mentality of the VRML 2 standard creators. The Living World's task force is developing a standard that encompasses how an author can identify shared behaviors and multi-user ability within the VRML scene graph file itself. Then, using the standard, the multi-user server developers can create a standard compliant server and multi-user functionality will be assured by the world author without specifically testing the world on each server. In the long run, if the standard is written well, Living Worlds could easily obtain the success that VRML 2 is currently enjoying. In fact, Living Worlds may become the cornerstone for VRML 3. Such a standard is useful if a united, connected, cyberspace is to be provided by many different interests.

In the short run though, using an external interface from the VRML 2 browser to other programs appears to be gaining a ground swell of interest. The Java programming language is meeting the needs for other Web based technologies and is being used for creating object behaviors in a VRML scene [19]. The Web is such a dynamic and ever-changing medium that developers are bound to keep pushing the technology through their own creative client programs and server technologies. An external interface in the VRML viewer client lets this rapid development process happen. Multi-user 3D world technology can improve rapidly because the five component technologies can be worked on separately by specialists and an improvement in any one of the world model builder, text/voice communications, Web browser, server, or client/server communications technologies improves the whole process.

4 INTERFACE STRATEGIES FOR A 3D COLLABORATIVE WORLD

Computer users have had the benefit of evolving human-computer interfaces ever since the first command structure was created. Human beings communicated with the first computers by way of switches and lights on a control panel. Soon thereafter, a terminal was attached to the CPU in order to provide a more dynamic interface. Users then could type characters on a keyboard and see the characters echoed to the terminal screen. Certain combinations of characters communicated a user's request for the computer to perform a function on the user's behalf. The results were then communicated back to the terminal when appropriate. Menus soon followed that allowed a user to pick a command from a list of possible choices without having to type the command.

As computer memory became less expensive and new monitor technologies were developed, computer systems could afford to express pictures on the monitor screen instead of simple characters. A pointing device was added to the user's arsenal of input devices and he or she used it to more freely select pictures and text on the screen. Whole new graphical user interfaces (GUIs) were developed that consisted of popular user controls, often called widgets. These widgets became integral parts of computer operating systems.

Because these GUI controls were so universal, the first desktop Virtual Reality (VR) systems used them for their human-computer interface. In fact, all the multi-user world clients discussed in Chapter 3 continue to show significant use of the GUI controls popular in 2D graphics. There seems to be incredible potential to invent new graphical interfaces that take advantage of the third dimension. Many 3D modeling technologies and 3D virtual worlds provide a third dimension in which interface components can be designed and implemented. This chapter looks at the traditional 2D GUIs and discusses possible new interface strategies for communicating with 3D multi-user worlds.

4.1 2D GUI Interface Controls

In a 2D world, interface components usually occupy a fixed location on the screen when enabled. This allows for inter-program consistency and easy remembering for the user. The evolution of interface components has always been toward easier use, less physical movement of the pointing device, and simpler task repeatability. As computer programs have become more complex with more available options to choose from, the GUI has

become the subject of much conversation. 2D GUIs have been evolving toward simplification and reduced size. Small size is important in 3D environments as well in order to allow the user to see as much of the world as possible.

Initially, GUI controls consisted of check boxes, option button groups, buttons, text boxes, and list boxes. These objects are the cornerstones for human-computer dialogs in a windowed operating system. A GUI programmer bundled the controls into different pop-up windows that appeared to respond to the user's actions. These same 2D GUI controls continue to be used heavily today in computer programs as well as on HTML-based Web pages. The HTML specification includes standard input attributes for check boxes, option buttons, single choice list boxes, multiple choice list boxes, text boxes, and simple buttons. Many list boxes take advantage of a drop-down feature where the user only sees the list of available choices after selecting a drop-down symbol to the immediate right of the box. Drop-down list boxes save precious screen real estate.

In an effort to cut down on the need to move the pointing device extensively, short-cut menus have become popular with the introduction of the Windows 95 operating system. In Windows 95 software, short-cut menus become available when a user right-mouse clicks on an object on the screen. The menu includes all context sensitive choices available based on the current state of the software in memory. If the choices are extensive, a cascading menu structure is used to organize the user's path to the desired option. Short-cut menus save pointing device movement since the menu appears where the user clicks instead of requiring movement to a specific menu or toolbar location. They also eliminate unnecessary memorization of menu choices and toolbar icons as all appropriate menu choices appear with the pop-up menu.

Lately, more and more computer programs and Web pages are taking advantage of defining multiple areas within the main application window. Such areas are called frames or panes. Frames and panes provide independent navigation by the user in order that multi-purpose application tasks can be better distributed for user control. Multi-user 3D world clients are a good example of a multi-purpose application. In a multi-user environment, the user participates in chat, world navigation, and options selection simultaneously. With a frame environment, the option selection decision is always on-screen independent of where the user is in the 3D world or the current status of her chat sessions. Each frame can contain the traditional 2D GUI controls to allow the user to interact with the frame's activity. Often, frame size is within user control.

4.2 New Opportunities with 3D

In 3D applications, the user is usually an active participant in the 3D scene and has a specific location and viewpoint in 3D space. This provides an opportunity for new human-computer controls. Traditional 2D GUI interfaces place the controls on the computer screen that is a barrier to the user's actions. The user can't move forward into the screen. With desktop 3D interfaces, the goal often is to reduce the user's sense of the computer screen in order to help immerse the user in the computer program. The user is afforded the ability to move forward and backward relative to the screen. Using the popular 2D GUI controls and placing them on a fixed screen location reduces the user's sense of immersion in many cases. In 3D environments, 2D GUI controls which are fixed in location change size based on the distance of the current viewpoint from the control. Placing 2D GUI controls in fixed locations in 3D space limits their availability to a subset of possible viewpoints. Is there a useful intermediate mode somewhere between always being on screen and fixation in a 3D location?

Also, in a 3D interface, the controls themselves can be three dimensional. Door knobs can open doors, file cabinet drawers can organize documents, and steering wheels can control vehicles. These controls are easily recognizable from many viewing angles. Traditional 2D GUI controls are flat and not easily manipulated from angles that are not perpendicular to the user's line of sight. Besides, traditional 2D controls are not often encountered by a user in non-computing activities in the real world. They are constant reminders to the user that they are not immersed in a real situation but, in fact, are being provided a computer simulated virtual reality.

Current desktop multi-user 3D world clients are not considerate of an immersed collaborator. The clients take advantage of the best known 2D GUI controls and frames. In theory, this provides the user the best chance of immediately using the software effectively based on his or her accumulated knowledge of word processors, spreadsheets, and Web pages. But, if provided the same interface while immersed in the world with an HMD based VR system, the user would find the interface distracting. Providing a different interface for non-desktop users means retraining for the desktop user who for the first time gains access to VR immersive technology. There exists an opportunity to create one interface that works well for both desktop and sight immersed users.

3D environments provide the ability for a user to have virtual hands and feet that are not necessarily within the field of view at any time. A 3D interface can take advantage of that fact by placing sophisticated controls in the hands of the user and then giving the user the ability to move his or her hands up within view or down out of view. Perhaps even better is the opportunity to let a user choose from different input devices that are made available in the world. The user could choose their input devices and controls by picking up the objects they want to use. There is no reason a user could not choose where to put their controls. Initially, while learning, a user could put their controls (or help information) in a place where they are always within view, such as dangling from the front brim of a virtual hat. Then, they could move those objects to their virtual hands to be brought within view as necessary.

4.3 Today's 3D Multi-user World Interface Solutions

As discussed in Chapter 3, the client is usually responsible for handling world navigation, text or voice communications, and personal user preferences. Each of these components requires some interface design. The following section reviews choices made by the 3D multi-user world clients available today for download over the Internet.

4.4 World Navigation

A virtual collaborator has six directions available to her when navigating a virtual world: up, down, forward, back, right, and left. In many worlds, the up and down directions are not available since the user is not intended to leave the ground terrain. In many clients, such as Sony Corporation's CommunityPlace, blaxxun interactive's Passport, OZ Interactive Inc.'s OZ Virtual and Netscape Communication Corporation's Live3D, forward, backward, right and left movement is initiated with the arrow keys on the user's keyboard or user's mouse movement when the user is in a walk mode. The up arrow or mouse movement away from

the user navigates forward in the world, the down arrow or mouse movement toward the user navigates backward in the world, the right arrow or right mouse movement takes the user to the right in the world, and the left arrow or left mouse movement takes the user to the left in the world. In SGI's CosmoPlayer VRML viewer, these movements are available when the dashboard has been turned off. The dashboard contains visual navigation controls that require mouse interaction in order to navigate the user. The dashboard is a user preference that can be toggled on or off by selecting the Show Dashboard item on the short-cut menu that appears when the user right mouse clicks anywhere in the world.

When navigating with the keyboard, the user is immersed visually without any artificial cues in the world. When navigating with a pointing device such as a mouse, the user sees a pointing device cursor that leads the user in the direction she is traveling. The mouse does allow more control as the user need not travel in a pure east, west, north, or south direction, but instead can move at any angle along the ground plane by moving the mouse in that direction. Sony Corporation's CommunityPlace viewer shows a tail connected to the cursor that defines the user's current direction.

Vertical movement while in walk mode varies substantially between viewers. Live3D and CommunityPlace use the CTRL key on the keyboard as a request for vertical movement. When the user holds the CTRL key down, the up and down arrows on the keyboard and forward and backward mouse movement move the user vertically instead of forward and back. CommunityPlace also provides a jump button that takes the user a certain distance above the ground plane and changes the viewpoint orientation to face the spot from which the user jumped. OZ Interactive Inc.'s OZ Virtual viewer requires a user to move off of the ground plane in order to fly vertically. The user then can move vertically with the pointing device and keyboard arrows. To enable this intuitive movement, the world designer must be sure to leave gaps in the ground design from which the user can get air bound. CosmoPlayer invokes vertical movement through its dashboard. The right-most control lets the user move up and down when selecting it with the pointing device.

As these viewers get more sophisticated, they are expected to improve on automatic terrain following algorithms while in walk mode. With terrain following in place, the user uses the keyboard arrows or pointing device to move forward, back, right, and left and the viewer follows the ground plane defined as the first geometry that exists directly beneath the user's virtual feet. OZ Interactive Inc.'s OZ Virtual viewer currently does a realistic job of terrain following when the gravity option is turned on by the user.

Virtual world clients typically allow the user to change the navigation mode in order to traverse the world in a different manner. Netscape Communications Corporation's Live3D offers five explicit modes: walk, look, spin, slide, and point. These modes are available from a simple on-screen menu in the lower left corner. The user clicks on the menu to select the mode of navigation she prefers and that mode is active until another is selected or a world is entered that modifies the mode while loading in the computer's memory. The look mode allows the user to move the viewpoint orientation without moving her location or changing her current feet plane. Keys and mouse movement are used to look right, left, up and down. Spin mode allows the user to spin the world vertically or horizontally, but not around the forward-back axis. Spinning changes the user's feet plane to reflect the change of scenery. Slide mode uses the arrow keys for movement left, right, up, and down. Point mode allows the user to click on an object in the distance and allow the viewer to select a viewpoint that brings the user closer to the object selected. These five modes allow for great flexibility of navigation for the user.

In CosmoPlayer, navigation modes are changed through the short-cut menu made available with a right-mouse click anywhere in the world. Navigation mode changes change the controls available on the dashboard. OZ Interactive Inc.'s OZ Virtual viewer allows the navigation mode to be changed from the menu bar that is part of the window frame. Navigation modes are not as explicit as in Live3D's viewer as the user turns on and off different attributes such as gravity to create the mode desired. CommunityPlace provides different controls on a picture based control panel. These controls allow a user to choose actions typical of different modes such as pointing or sliding without explicitly changing the current mode.

All the viewers allow the user to enable and disable a default light in the world. The light, called a headlight or headlamp is a directional light that emanates from the user's forehead. Some viewers disable the default light when the world dictates. In all the browsers reviewed here, default light choices are made from a menu. CommunityPlace and OZ Virtual include the default light option from an available menu bar menu. Live3D and CosmoPlayer include the option in the short-cut menu provided the user when he or she right-mouse clicks anywhere in the world. Live3D additionally shows a headlight option on the navigation menu seen in the lower left of the screen at all times.

Collision detection defines whether a user can walk through solid objects when navigating in a virtual world. Collision detection can be enabled or disabled on the same menu as the headlight choice. Yet, Live3D does not make the current collision detection decision explicit on its on-screen menu.

Current navigation speed is typically chosen by the viewer based on the complexity of the world model. Yet, the user modifies the speed by her actions. CosmoPlayer provide an explicit menu choice on its right-mouse click shortcut menu. A user can choose slow, normal, or fast navigation speed. In the others, speed is derived by the speed of any pointing device movement and the length of time a keyboard key is kept depressed. A user accelerates as a key is held down until some terminal velocity is reached.

A note of interest is the lack of 3D objects used for navigation assistance. Only CosmoPlayer uses 3D objects in its interface, providing 3D objects on its dashboard.

4.5 Integrated Communications

For worlds with voice communications, the communications interface can be quite simple. Both OZ Interactive Inc.'s OZ Virtual and OnLive! Technologies Inc.'s Traveler viewers provide nothing more than volume control and a mute button. Volume is adjusted through two simple arrow buttons where a click on the up arrow increases the volume by a fixed amount and a click on the down arrow decreases the volume by a fixed amount. The user clicks multiple times to change the volume dramatically. A user clicks on the mute button if he or she does not want to send or receive any voice communications. Then, the client has more resources available to world rendering.

For worlds with text chat communications, the text environment is controlled by the user through adjustable multi-line text boxes and a fixed single-line text box. The multi-line text boxes show the user the chats to which he or she is active. In Intel Corporation's, blaxxun interactive's, OZ Interactive Inc.'s, Circle of Fire Inc.'s, Worlds Inc.'s, and Sony Corporation's multi-user clients, the user is always privy to a global chat and then made privy to any other chats in which the user gets involved. Each of these chat boxes can be resized by dragging the chat box border with the mouse. blaxxun interactive provides an attractive point and click menu from which a user can jump between chat sessions. The user then need only make a single multi-line text box big enough for comfortable reading.

The work by blaxxun interactive provides another communications vehicle, the business card. Each user can save a local business card with information about themselves in the real world. Then, through a menu, the user can request and send business cards to other users.

In no client that I visited did text chat communications take advantage of the 3D world for interface presentation. I found the chat environment to be very natural and intuitive given my background with text editors and word processors. Only in Circle of Fire Inc.'s Active World did I encounter the concept of a homestead where users could put their own pictures and information in the world. Users could also add a mailbox to their homestead on which other users click to send email to that homestead's creator. A next step would be to add a telephone to each homestead in order to initiate voice communications (via telephone or, if active in the world, a voice channel).

4.6 Personal Choices

In all the clients I used for multi-user communications, options are presented to the user via traditional 2D GUI controls and context sensitive use of the right mouse button. Controls are provided via HTML controls within frames or Java based interfaces existing within an HTML frame. I found no new, innovative uses of 2D controls and thus found making option choices very intuitive for anyone skilled in using popular graphical operating systems or Web browsers. I found much innovation in what world options are made available, but not in the controls that allow choices to be chosen.

4.7 Immersion Strategies

For an immersed collaborator, the screen no longer exists. For now though, Internet connected, multi-user world collaborators are more often not immersed and the screen is used extensively with the interface design. Multi-user clients take advantage of a typical collaborator's computing experience to provide typical mouse-based controls. Menus, GUI controls (text boxes, check boxes, drop down lists, etc.), toolbars, toolboxes, palettes, dialog boxes, and combination widgets are all used to let the collaborator interact with a 3D multi-user world. The client developer usually uses a multi-windowed approach to organize the 3D world navigation, communications, and personal options selection interfaces.

Traditional desktop interfaces undoubtedly provide ease of learning to first time users, but don't support a natural environment for an immersed participant. 3D world interface designers have an opportunity to design with both the immersed and desktop collaborator in mind. I believe users who immerse themselves in a 3D world for the first time will have a more pleasant experience if they can interact with the world in a manner similar to how they have been interacting through the desktop. Yet, immersed behaviors need to be natural in order to get a user to believe they are physically in the world and not in a computer simulation.

4.8 Discussion

The RSV tool is available for game players and response coordinators to use to evaluate performance of first responders during a simulated emergency response game play session. Evaluation requires an evaluator to develop the metrics by which an emergency response effort is considered successful. Metrics vary greatly by the different constituents in a community. Some organizations in a community have significant investment in physical assets. Some organizations, like a museum, may have fewer assets but the assets may be of priceless value due to age or significance to human culture. Most constituents agree on the priority of saving human life, but don't agree on the relative priority of saving pets or livestock. Where does the cost of gasoline used to transport responders and resources fall within a scoring metric?

The RSR tool allows a scenario developer to determine a scoring algorithm and we show the team score at all times based on this algorithm at run-time. An analyst can refine the scoring algorithm by analyzing its impact on performance in order to determine its effectiveness in generating desired behavior from game players. Alternatively, an analyst can start with the RSV tool and find an example of team behavior that appears to be most successful and then use that example to build a scoring algorithm based on seeing scenario-appropriate behavior. Ideally, the RSR and RSV tools can be used in unison to iterate upon a better scoring algorithm with which a player can play with software-based agents and get a sense of how well he or she is doing.

We build the RSV tool to support one basic metric of insight generation – the more insights that are generated from interacting with the RSV, the better. Although this is a simple metric, it is consistent with goals of the visual analytics community in general. The RSV should allow anyone to get a deeper sense of how an emergency response effort performed just by interacting with simple widgets that accumulate value in their coordination in groups.

Based on observing users use the RSV after having gained familiarity with the RSR through repetitive use, we hypothesize that a single scoring algorithm is not sufficient for building an optimal perspective on an emergency response effort to any scenario. Instead, a visual tool like the RSV tool lets an analyst discuss a response team's performance with changing metrics associated with changes in the nature of the unfolding scenario being analyzed.

5 MY COLLABORATIVE WORLD

All of the 3D collaborative, Internet based worlds that I visited and spent time in (OZ Interactive Inc. worlds, Active Worlds from Circle of Fire Inc., Worlds Inc. worlds, OnLive! Technologies Inc. worlds, Moondo worlds from Intel Corporation, People Space worlds from IBM, Circus World from Sony Corporation, Pointworld and other blaxxun interactive worlds), use a reasonable architecture of landscapes and buildings for the 3D world. All presented me with a world where I could quickly orient myself, put the territory to memory, and walk about to visit different locations within the world. Through my cyber-journeys,

I came to the conclusion that many people are working on creating attractive, functional 3D architectures. I was able to talk to enough trained architects who were studying the architecture of cyberspace to believe they will continue to build better buildings in cyberspace as time goes by.

Instead, during my cyber-visits, the lack of interesting things to do in the worlds disappointed me most often. I could walk around in a functional and attractive world and communicate with others, but that was all I could do. The VRML 2 standard specifies how to include interesting smaller worlds within the confines of a larger world. I wanted to work on the smaller worlds that could then be contained within the architecture of any larger VRML 2 world. I would leave the larger architecture for others to design.

5.1 Objectives for My World

After a lot of thought and learning (I designed a Chinese checkers world, billiard world, and kaleidoscope world, all of which could be placed on any flat surface in a larger VRML 2 world), I found my own creative concept that I was interested in implementing. I decided to build a virtual world that would require and emphasize collaborative and competitive behavior and provide both as alternating goals for a participant. The world would be available for placement on any flat object such as a table or floor. I defined the following critical success factors for the world:

- Demonstrate collaborative behavior and contrast it with competitive behavior
- Demonstrate an appropriate use of mixing VRML and Java technologies over the Internet
- Provide an enjoyable experience to attract prolonged attendance in the world
- Provide component parts that could be put together by participants in unexpected ways
- Demonstrate the attributes of a complex system

This chapter provides an overview of my motives behind each critical success factor and then presents the design high points of the world.

1. Demonstrate collaborative behavior and contrast it with competitive behavior.

I understand why people have such a difficult time defining collaboration. Everyone has a different image in their mind when they think of the word *collaborate*. I wanted to create a world that provided alternating reward structures: first for competitive goal reaching, second for team work, and third for collaborative goal reaching. I hoped that the different objectives would make participants behave differently. Those different behaviors would be called competitive, teaming and collaborative. By participating, a participant would be able to experience the three behaviors and, hopefully, as a result have more insight into all three.

I experimented with ways to create such a reward structure. I decided to model the goal structure after an organization attempting to gain market share. In an emerging market, there are at least three ways to gain absolute market share (the number of customers) for a producer. The first way to gain market share is by taking customers away from other organizations (competitive). A second is by working with another producer to jointly grow their market share (team). A third way to gain market share is by promoting the whole industry to grow the absolute size of the market yet maintain the same relative share (collaborative).

Much has been said and written about the merits of competition in securing market share for a producer. Less has been documented about the merits of collaboration. Innovation in the hamburger industry and car industry can appear to be somewhat collaborative when each key producer adds new features to the basic product that drives up the demand for the product as a whole (versus other foods or forms of transportation). Collaboration can also be more direct. For example, in a collaborative manner, the organizational members of an industry can work together to increase the demand for their product or service. For example, the national dairy council promotes milk for all producers of milk through their *Got Milk?* campaign.

Of all the industries I considered, the oil industry maps more directly to the structure of my world than others I explored. In the late 1800s, four large oil conglomerates (called trusts at the time), dominated oil production in the world (Standard Oil in the US, the Rothchilds in Europe, the Nobels in Europe, and a group of Russian producers). Another participant, Royal Dutch, joined the battle shortly thereafter as the Russian producers began losing market share. There was much direct competition and joint ventures between two producers were formed from time to time, yet the whole oil industry worked together to promote oil over other forms of energy [22].

For a high-tech product or service, the marketplace may not be aware of a new product or service. Once they become aware, they still may not understand why they would be interested. In fact, even the venture capitalists who are raising money to support the product or service may not be convinced of its viability. In those cases, collaboratively building awareness of the product or service by the producers provides a great return down the road. Taken a step further, such collaborative gain can include standardizing the technology in order that it works with other technologies already available to and owned by the market. VRML is a technology following that path through the efforts of the VRML Consortium [23]. The venture capitalists have provided funds for VRML development with strings attached. If the market is not grown to a certain level by a certain date, no additional funds will be forthcoming. So, there is much interest in standardizing VRML by the technology providers.

My world gets people to think about how to collaborate in the extreme: with people they neither know nor have ever seen. I figure such ability to collaborate would be extremely useful if the Web is to be used effectively for its new, unique ability of connecting any computer in the world with any other computer at any time.

So, in my world, a participant competes to reach certain physical locations in the world before others get there. These goals are represented by charcoal gray filled circles that lie flat at random locations on a board. During a competitive round, a participant attempts to win market share by reaching the goals first. Yet, in the next round of participation in the world, participants succeed by working together collaboratively and by attempting to get at least one participant to each location in the shortest period of time.

If there are enough participants, a third situation can be added where participants work as teams to confront other teams. If there are not enough participants, computer simulated participants could be added to provide competition or collaborative partners. Team competition is a behavior required by many board or card games such as bridge. My world appears similar to a game, but abstractly represents all virtual worlds that attempt to build a community through each individual's actions.

2. Demonstrate an appropriate use of mixing VRML and Java technologies over the Internet

In order for my world to meet the basic environmental constraints of the technology I wish to use, I needed to create a communication system that could overcome the latency of the Internet where reliable service is not guaranteed. I would also need to implement a solution that kept up a reasonable frame rate. By keeping my world to a standard implementation of VRML and the Java External Authoring Interface (EAI), I was assured certain inefficiencies in local processing within the Web browser. Yet, I believed it was important for me to use standard VRML in order that my world could participate as a part of a larger virtual universe at any time. In fact, I believed that my world could easily be put into a room of a larger environment such as the virtual 3D chat environments I comment on in Chapter 3.

I believe that the documented constraints of client/server applications that use VRML on the Internet are not limiting in many cases. So, to prove that, I wanted my world to be completely realized as a VRML world with Java scripting and Internet message passing using the hypertext transport protocol (http). In my world, the latency of the Internet would actually be incorporated into the design. I believe there are many useful multi-user applications where the response to a participant's actions need not immediately take effect. For example, in simulating market dynamics, a delay is quite realistic. When an organization puts a market strategy into effect, it may take weeks or months until the market begins to react. Many games of thought also do not require immediacy in terms of relaying moves from a mover to all other participants. To learn the technologies of VRML, Java scripting, and client/server communications, I created both a Chinese checkers world and virtual billiards table that in no way required a rapid response mechanism. Both Chinese checkers and billiard table worlds implement a turn-taking protocol that is consistent with their real world protocol.

Instead, the server that connects participants in my world need only be able to conservatively estimate a latency that is a worst possible scenario and then add a certain amount of time to it to find the least acceptable latency for the world. The server then need only assure participants that they have not chosen a latency that is shorter than the least acceptable latency. I intended to build that logic into my world server. Should the minimal set latency ever be violated, the server would simply stop the simulation until any latent messages could be delivered.

Of course, each event that needs to be coordinated between participants must be assured to take place at the exact same frame of the simulation. My world has an event queue built into its design to make sure each event is enacted at the same point of the simulation (though not at the same absolute time). Since all other actions are encoded within the physics of the world itself, and since each participant has that code loaded locally within their Web browser, only the events that are queued could possibly cause the different worlds to get out of synch.

All messages that get passed between participants in the world during an active simulation contain a future framestamp (a term I use to refer to the specific frame number in a simulation in which a certain state change becomes active). By not enacting a message until every participant had received it locally (before the agreed upon worst-case-plus latency), I would assure that the world state would always be the same for each participant.

3. Provide an enjoyable experience to attract prolonged attendance in the world

By prolonged attendance, I mean keeping each participant's interest during each session as well as an interest in returning to the world time after time. For me, keeping a participant active in a current session until the session was over was very important. I expect participants would be disappointed if they lost the other participants in the simulation during the experience. Although I envision someday having an automatic process replace the expiring participant, I had no doubt that that would be a less desired state if a sense of community surrounded my world. Unless, of course, the participant was detrimentally and purposely disrupting the natural tranquility of the world. In any case, the basic experience in the world had to be an attractive one.

To keep participants coming back, I figured I would need to provide a potentially changing experience in order that a participant was never experiencing the exact same experience each time. I also believed that I could create a sense of community where participants would come to reunite with participants they had enjoyed participating with in the past.

During my readings on complex systems, I realized that I would only need to provide a few different components in the experience that changed between visits to make the experience seem significantly different. I thought about how different two separate games of chess become, just by altering the first couple of moves. Participants would be making moves in my world. Yet, I wanted to go a step further and allow the rules to change between visits. With changing rules that could be defined and agreed upon by the participants, I wanted my world to be flexible enough to provide a wide range of experiences.

As I document in Chapter 6, I intended to test a hypothesis about prolonging interest in my world. My world would have enough flexibility that participants could build the world themselves and change the rules often during their participation. My hypothesis is that participants will be more interested in a world where they create the design and choose the rules.

Whether or not my hypothesis is true, I wanted to build a world that had the flexibility to change often should participants desire variability. In the extreme, I envisioned participants emailing me with new design objects, physics, and rules that I could make available in the world with a quick turnaround. For those who preferred to participate in a certain fixed configuration, I envisioned creating a library of successful variables that any group could agree to use in a particular simulation. In other words, my world would cater to both types of participants. Given my world architecture, it would be easy to store and access successful combinations of design, rules, and events.

4. Provide component parts that could be put together by participants in unexpected ways

I wanted my world to follow a serendipitous evolution of its own. I would provide objects and actions for the world and the participants would work to use them in ways I could not have anticipated. By providing much power in the hands of each participant, each participant's thought process would be reflected in the experience in the world. Even the component rules would be able to be added, eliminated, or altered upon participating in the world.

Since my component parts worked so well with an object oriented programming style and, since I would be using Java (an object-oriented programming language) to tie together the actions of the objects in the world, I figured I could provide individual encapsulated objects (visual, behavioral, and rules-based) that could interact well with each other. The physics of each visual object would be embedded within that object. There would be little that was pre-determined about how the world played out. My world would enact the same kind of collaboration among its code objects that it would be asking from its participants.

Since each participant would make independent decisions about what to do in the world and each participant would not be able to see what others were doing until they did it, I expected to find a strong independent behavior set coming from each participant.

5. Demonstrate the attributes of a complex system

With all the interesting thinking I had done as a result of reading about complexity and chaos, I wanted participants to get the same sense of amazement about complexity as well, but not as a frustration. Hopefully, my world would contain what looked like a manageable number of variables and a minimal number of participants in order that a participant would feel quite powerful in his or her ability to influence the world and others. Yet, by participating, a participant would soon realize how complex the environment was because others were independently trying to effect the outcome of the world at the same time as the participant.

The lessons from my world would be lessons about life. In many cases, we are not in control of how things turn out. We are only in control of our attempt. It can be fun to try even if the results are not what are expected. A virtual world should be able to be a place to learn and have fun without the goals needing to dominate the experience. I want the world learning to be about complex systems and an individual's ability to control his or her environment. Even in a complete and best collaborative effort (if there is such a thing), a group is not assured of reaching a goal. Yet the experience of being part of a group can be quite fulfilling.

Computer simulated worlds have a great benefit over physical worlds in that they can easily be made self-documenting. As part of the experience of building this world, I hoped to be able to save certain worlds and their events that best demonstrated emerging behaviors that were being experienced in the world. I hoped I could categorize certain people's behavior and find a way to illustrate collaborative behavior to people who were struggling to define it for themselves. A shared world could be experienced by another group of participants even if it were designed by a different group. A new group could try to outperform a previous group with the same world conditions.

5.2 Design Overview

With these goals in mind, I began coding my world. Much has been discussed about the architecture of virtual worlds in technical papers and at virtual reality conferences. My world would not need to address many of the issues of world architecture as I decided that my world would be a flexible world that could exist within the confines of a larger virtual world. My world could exist in a room of a virtual castle, office building, or shopping mall. Given the structure of VMRL, my world could easily be put into any other virtual environment.

All the popular 3D multi-user worlds I reviewed in Chapter 3 provide a large, organized world for multiple participants to explore. Yet, those worlds lack things to do while in the world. My world is a world that could be placed in any one of those larger worlds to provide a flexible context in which participants can share an experience. Although I provide specific objects and objectives for my world, it is the desire to produce a flexible and interactive design environment that has driven my work.

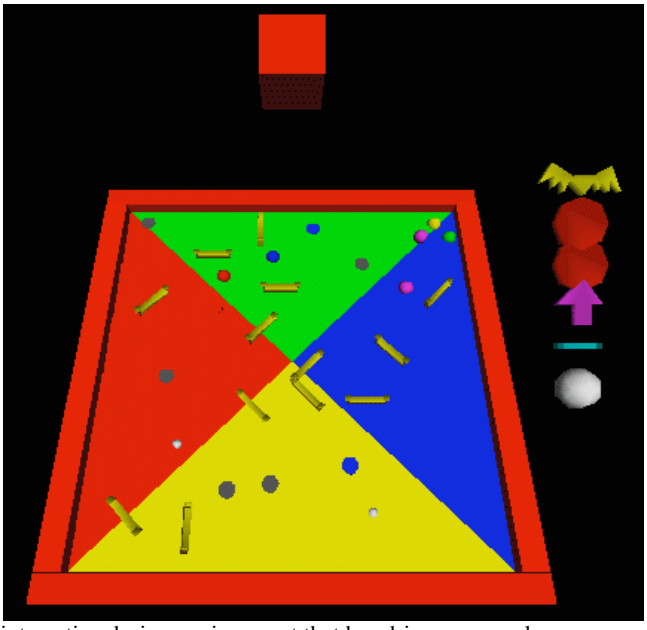

Figure 5.1 A Screen Shot of Marbles World

My world does have a context that is outlined in detail in Appendix A. A screen shot of my world in action is presented in Figure 5.1. I provide a quick summary of how my world behaves here. A participant in my world first clicks on check boxes to negotiate the rules for the world session with the other participants. The rules cover how certain objects on the screen behave, determine values for the overall physics such as a gravity coefficient and the mass of the marbles, and decide what constitutes success (including the competitive v. team v. collaborative decision). Once the server delivers the visual goals for the next round, participants together design the board on which their marbles will roll. Participants design the board by taking turns in dragging and dropping available design objects. After they design the board, they choose from a palette of available effects to add to their arsenal for the active simulation round. Then, the marbles begin to roll and the world comes to life. Participants click on the board to affect the slant of the board which dictates the direction the marbles move. Participants also click on the effects in their arsenal to control their marbles within the embedded physics of the world. The active simulation

continues until the agreed upon condition for ending is triggered (for example, all goals have been reached by at least one participant). Then, the rules setting starts again. The session continues for a fixed number of rounds.

To code for such a world, I had a lot of architectural decisions to make. Client/Server architecture provides great flexibility in placing functionality at either end of a socket connection. I experimented with placing different functions at both ends. Over time I settled on an architecture that worked well enough to keep the frame rate acceptable on each client yet also kept the participating clients in synch. My client/server architecture appears as seen in Figure 5.2.

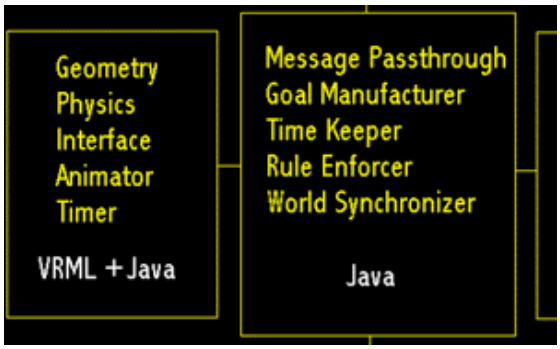

Figure 5.2 - My Client/Server Architecture

Basically, with the architecture in Figure 5.2, the client is burdened with most of the processing. The client manages the geometry of each visual object in the world, drives the behavior of each object from its embedded physics, provides event handling routines for each event triggered by the participant, calculates and presents the animation of all objects that move and collide with other objects, and maintains a constant speed for each animation frame loop.

In comparison, the server's processing load is light. The server receives messages from clients and passes them through to the other clients involved in the simulation. The server also randomizes the placement of any goals for the simulation and forwards their location to each client. The server keeps its own timer to make sure participants are responding within any established time limits. The server enforces any rules established by the actions of the clients, effectively dropping certain client messages that go against the established rules. The server also receives timing reports from each client and changes the animation frame loop time for clients that are running faster or slower than the average. The server is responsible for keeping each client simulation in synch.

5.3 Client

My world is basically a coordinated effort between various extendible classes that connect to a standard VRML 2 file through a standard Java External Authoring Interface. I present an overview of the key Java client classes and their responsibilities in Table 5.1.

Table 5.1 Alphabetical List of Project Major Classes

Animator

The Animator class runs the frame loop that drives the simulation. The Animator class handles messaging between all other objects in the world by performing update and collision management given the current state of the Rules class.

Effects

The Effects class defines special abilities of marbles that participants can use to direct their marbles during the active phase of the simulation. For example, the effects class defines an anti-gravity effect a participant can use to move her marble against the pull of gravity. The Effects class can add new effects to itself easily at any time. The current state of an Effects object determines which effects are made available to the participants in the world.

MainWindow

The MainWindow class contains a question text box, answer text box, and 10 checkbox/label pairs that can be used to ask questions to and accept answers from participants. The answers from these questions change the state of the Rules object. The MainWindow class can easily add new questions to itself at any time. The current state of a MainWindow object determines which question, if any, is currently being asked.

Marbles

The Marbles class, inherited from the Applet class, sets up and maintains the connection between the VRML 2 viewer and the Java virtual machine. The Marbles class also starts the Timer, PortToServer, and Animator threads.

Physics

The Physics class defines the behavior of the pieces that can are used during the active phase (post-design) of the simulation. For example, the physics class contains the behavioral logic for a pendulum. The Physics class can add new physics to itself for pieces easily at any time. The current state of a Physics object determines when and how objects can move during an active session of the world.

Pieces

The Pieces class defines the 3D geometry of the pieces that can be used during the design phase of the simulation. For example, the pieces class contains the geometry for a vertical barrier. Participants can add one or more vertical barriers to the world during design. The Pieces class can add new pieces to itself easily at any time. The current state of a Pieces object determines which visual objects are made available to the participants in the world.

PortToServer

The PortToServer class connects a client to the world connection server and handles incoming messages from the server. The PortToServer class can easily add new message handlers at any time. The current state of a PortToServer

object determines the effect incoming messages have on the world.

Rules

The Rules class contains key attributes that define how the world behaves. Rules can be changed by participants in the world. For example, the Rules class defines how the slant queue loads its slants (randomly, through participant turn taking, or through participant first come, first serve). The rules class can quickly be extended at any time to add new rules to the world. The current state of a Rules object dictates how the Animation class behaves during a simulation. The Rules object changes its state based on how participants answer questions asked by the MainWindow object.

Ticker

The Ticker class controls the speed of the Animation thread by setting a sleep variable based on feedback from the server. The animation then sleeps for the appropriate period once per animation loop.

There are also smaller classes that perform very particular roles such as the Ball, Obstacle, Goal, Player, Arsenal, EventQueue, and SlantQueue classes. The Ball and Obstacle classes keep track of the objects that interact during the active phase of the simulation. The Goal class keeps track of each goal active in the simulation. The Player class manages the scoring for each participant. The Arsenal class manages the effects a participant has chosen for the next round. The EventQueue class manages upcoming effects and passes them to the Animator when the time arrives for an effect to become active. The SlantQueue class manages the upcoming slant orientations of the world's board and passes them to the Animator when the time arrives for a slant to become active.

5.4 The Server

With my client design falling into place, I figured my world server, written in Java, would be as simple as possible. I decided to start with a simple message passing server that would do nothing more than pass messages from one participant to another. The server would have no understanding of the messages it passed and all incoming and outgoing message logic would be contained at each client. As I worked with my design though, I found it better that the server be aware of the speed of each participants simulation, be responsible for creating the physical goal locations for each simulation, and keep track of which participant was which marble. Given all possible configurations, my server was relatively simple. I started with base classes that handled socket creation and maintenance and added code specific to my world. Yet, the code I added would be appropriate for many other shared virtual worlds. In all, the server Java file contains two classes that together connect all the participants together in a shared 3D world simulation.

Appendix A contains a primer I created that new users can use to acclimate themselves to how the world operates. Appendix B contains my client code as it existed during my pilot test outlined in the next chapter (Chapter 6). Appendix C contains my server code as it existed during my pilot test. Appendix D contains my VRML world.

6 TESTING MY COLLABORATIVE WORLD

Having built my world, I wanted to run a pilot test that would test marbles world against my stated objectives. The pilot tests would be somewhat informal, yet give me some experience with running tests should a more formal experiment become warranted. I would ask the participants various questions about their interest in the world, get feedback about the technical aspects of the world itself, and ask about their desire to return to participate again. At the same time, I also wanted to obtain data from the participants that would hopefully support or refute a simple hypothesis. I considered a few possible pilot test goals such as comparing performance between anonymous participants and participants who met each other beforehand or comparing collaborative behaviors of participants before and after participating in the world. The architecture of the world, being so easily extended, opened a long list of feasible pilot tests I could run. The world could be modified with minimal effort to run many different pilot tests.

I decided to run a pilot test which would grapple with a question related to one I had read often in magazine articles about Virtual Communities: What makes people remain in and return to a Web site? Many articles propose that people like to visit Web sites where they feel like they are part of a community. The community can be as simple as those who drink the same cola, or as life enhancing as people who are fighting big government against a perceived wrong. The idea that people come to a virtual place where they feel comfortable with the other people who are there parallels a real world phenomenon that keeps people coming back to their church or university. I wanted to take the comfort idea in a slightly different direction and consider whether people remain in and return to Web sites because they have a say in what takes place at that Web site. Howard Rheingold's Electronic Minds Website was an example of a text-based Website where people visited and drove the conversation themselves. I visited Electronic Minds and opened a discussion on whether the analogy of an Information Superhighway is a good one for explaining the Web to the masses. I certainly was drawn to Electronic Minds because I had a say in what happened there. I quickly perceived myself as being a part of a community with the others visiting.

In terms of Internet based 3D environments such as my marbles world, allowing for audience participation is a newer frontier. I imagine a sandbox world where a participant can connect via the Web, bring his or her own visual objects into the world, and share an experience with others using the objects each participant brings. Or, I can imagine a gallery world where participants bring their own works of art into the world and work together to present them in the most attractive and functional manner. Gallery world's gallery could even be built from scratch by a participant base given appropriate tools. Or, I can imagine a virtual golf world where participants build a golf course before playing golf.

These three imagined worlds can all be feasibly built using VRML, an external authoring interface and a Java server. The worlds I suggest in the last paragraph each have different levels of participation. A sandbox world could have virtually no pre-set objective outside of sharing time with others. A gallery world might be quite a bit more organized. A golf world might follow very specific rules for playing golf. The questions that beg research are *how much freedom to Web participants want?* And

are Web participants more apt to stay in and return to Web sites where they have control over what takes place in the world?

The first question is very interesting, but I believe depends too much on individual preference to test with a limited subject group. The second question is interesting as well, but I think may have too broad a scope for a simple pilot test to provide meaningful results. The question I decided to investigate is whether Web participants perform better in worlds where they design the world and participate in the rules determination than those Web participants who just show up, read the rules, and participate in a pre-built, 3D world.

Since marbles world provides a library of objects, a library of effects, and a library of rules from which to define what takes place in the world, there is a opportunity to set up a pilot test between two groups: one group which together defines the rules, builds the board, and runs the simulation, and another group which just runs the simulation the previous group defines.

The hypothesis I offer is that the group that creates the world through its participants will perform better given the rules they choose than the group that just follows the rules and design set by the previous group. I am not sure of the full implications of those expected results, but I believe they are related to experiments run on a participant's sense of immersion in 3D worlds. Those experiments suggest that virtual participants feel more immersed in a world when they are interacting with it based on their ideas and actions. If the hypothesis holds true, Web designers would be supported in creating Web sites with more flexibility. If the hypothesis is negated, Web designers may well be justified in continuing to mass-produce more static Web pages designed to work well with even the first Web browsers and Web TV.

It seems to me that the Web becomes significantly less important if Web participants are not craving participation with others. Broadcast TV can deliver content to the masses effectively. The Web provides a possible direct connection between any two computers (or people) in the world. With the Web, there exists an opportunity to use technology in completely new ways. If people can use a Web connection to create something new of value, proliferation of the Web may be quite a wise investment. Since my hope through this project has been to become effective at using a technology that provides new and useful ways of collaborating electronically, this pilot test will also provide some initial feedback as to whether people would use the technology given the chance.

As a last paragraph before outlining my pilot test, I want to mention how communicating and sharing through a Website could dramatically change the way certain things happen in society. Today, many of the products and services consumers purchase in the marketplace are first introduced through a product development cycle that very often takes place within the walls of large corporations. Consumers get the opportunity to provide feedback to product development personnel through surveys and focus groups. Although the Web has made it easier for consumers to provide feedback, flexible Web sites could allow consumers to design their own products and then take them directly to a product or service provider for realization. Similarly, many games are created by game development companies that instead could be creatively developed and tested within the confines of a flexible website by independent Web participants. Product and game development worlds can be built along the same design philosophies I follow in building marbles world. The question is whether people would use them given the chance.

6.1 Preparing Marbles World for a Pilot Test

In order to prepare marbles world for the pilot tests I wanted to run, I decided to remove some of the code that had provided flexibility. I figured the results would be too hard to analyze if participants could choose from the full rules set I had created and tested. How could I compare simulations run in competitive mode and collaborative mode? I decided to limit the pilot tests to collaborative mode since I was most interested in collaboration through the Web anyway. I then decided to use a fixed set of objects, events and physics to cut down on the learning curve for participants. By discussing the upcoming tests with fellow colleagues, I came up with a fixed way to manage the slant queue. I decided the world would load initially with the first two slants already determined. I found that the group design activity was more interesting with the two slants already in the queue. Lastly, I spent time fixing the server code so that a specific list of questions would be asked to any group that was determining the rules. I then was able to focus my code test plan more specifically.

6.2 My Pilot Test Plan

I will find twelve or sixteen subjects who are willing to spend a couple of hours maximum in front of a computer monitor participating in a virtual world. Three or four subjects will participate at a time. For each group, I will give them written instructions to read and then spend time with them answering all there questions except questions about group strategy. Each group will participate in two simulations. The first time around, the group will collaborate in a world designed by a previous group. The second time they will choose the rules and physics for the world, design the world, and then run the simulation in a manner similar to the first time. All participants will work toward an objective of reaching a fixed number of goals in as little time as possible. After each simulation, all participants will answer a post-simulation questionnaire. I will then look at the differences between answers from design round participants and non-design round participants.

All participants will be able to read about the world before they participate and ask questions about any directions or procedures they do not completely understand. But, the participants will have no previous physical practice with the world. I will only require that participants have average or better mouse skills. My world will be fixed in collaborative mode in order that a group's success will depend on how quickly the participants get at least one marble (but any marble) to reach all the goals that appear in the world. The world will keep track of how many frames it takes to finish the simulation, as it also keeps track of the rules and design. Should the participants be unable to reach all of the goals within 2000 frames, I will stop the simulation and count the number of goals they were able to reach.

After their participation, I will ask each participant questions that rate their experience in the world as to how much they enjoyed the experience, how well they believe the technology performed, how likely they would return to the world, and how immersed they felt in the experience. The simulation will be considered more successful if participants enjoyed the experience, supported the technology, felt likely to return, and felt immersed in the simulation. My quantitative data will report on the performance of

the groups. I will introduce a scoring system to the world that will score points for getting to goals in the shortest time possible. I will compare the cumulative scores between the design simulations and the non-design simulations as a more objective comparison of the two groups in action.

6.3 The Post-Simulation Questionnaire

The questionnaire I will give to each participant after they participate appears as follows:

> Please answer the following questions on a scale of 1 to 7, 1 meaning you strongly disagrees and 7 meaning you strongly agree:
>
> 1. I enjoyed participating in the simulation.
>
> 2. I became immersed in the simulation with little awareness of other things going on around me.
>
> 3. I would like to participate again with the same group of participants.
>
> 4. I would like to participate again with a new group of participants.
>
> 5. I thought the technology worked well.
>
> 6. The interface was natural to use.
>
> 7. I felt like I was treated as an equal in the simulation.
>
> 8. I learned something about collaboration during the simulation.
>
> 9. I had more control of what took place than I anticipated.
>
> 10. I would like to have more control of what happens in simulation.
>
> Please provide written comments related to the following:
>
> 11. Please describe your frame of mind when you started the simulation.
>
> 12. Please describe your frame of mind during the simulation.
>
> 13. Please describe your frame of mind right now.
>
> 14. Please list any frustrations you experienced while participating in the simulation.

I ask question 1 because I believe Web visitors must enjoy participating in a world if they are to become part of its community for the long-term. To evaluate my world, I must determine whether participants enjoyed the simulation. I ask question 2 because I am interested whether my world was immersive or not. Immersive worlds are more successful than non-immersive worlds in keeping a community together and vibrant. I ask questions 3 and 4 since more successful communities have members who want to return to visit often. I ask question 5 to differentiate between disinterest from problems with the technology and the disinterest from the underlying idea for the community. I believe the technical problems can be easily overcome. Fundamental idea problems would be more difficult to overcome. I ask question 6 to grade myself on my interface design. I ask question 7 to confirm that a server is a fair and just facilitator of actions. I ask question 8 hopeful that participants will learn about collaboration during their participation. I believe they would learn even more if they participated in a competitive mode as well. I am not providing competitive rounds as part of the pilot test. I ask questions 9 to get a sense of whether expectations from reading my instructions coincide with the reality of the simulation. I ask question 10 to see how participants feel about not having as much control as many video games provide. I ask questions 11, 12, and 13 to get a sense of how participants' thoughts change throughout the pilot test. I ask question 14 to get feedback that could help me make the simulation better. I believe each question will provide valid feedback. I now run the pilot test and review the results in Chapter 7.

7 RESULTS OF TESTING MY COLLABORATIVE WORLD

This chapter reviews my overall thoughts from this Masters Project, my pilot test data, and considerations for more research. Through studying VRML 2, the Java External Authoring Interface, and client/server design, I came to the conclusion that two extremes of flexibility are possible in 3D virtual worlds. I could use the available technology to create a virtual chess board game whereby the server would enforce the official game of chess rules. Or, I could design a virtual world where nothing happens until participants bring their own objects into the world and make something happen. The latter might be a world that is appropriate for learning a foreign language as each participant adds the objects to the world that they feel comfortable talking about using the foreign language under study.

Between the two extremes of world flexibility, there exists a broad spectrum of possible multi-user world rule sets. I tried to make my world fall near the center of a flexibility scale. My world has a fixed context, but much flexibility around that context. I found participating in such a world quite enjoyable and more rewarding than some other experiences I had visiting existing 3D multi-user worlds. Building and participating in such a world brought me to consider some interesting questions and conclusions.

7.1 Overall Thoughts

After thinking about VRML, Java, 3D virtual worlds, and networking for twenty months, I finished the Marbles world project with some thoughts that I did not entertain when I began the adventure. The following section documents some of my more prevalent thoughts.

1.The External Authoring Interface works OK on Pentium class machines.

When I started building a client/server architecture for sharing VRML 2 worlds on the Web, I was not sure PC technology had advanced enough to be an appropriate platform for my world. I was extremely pleased with the potential that I saw when I placed my first virtual marble on a flat piece of wood and watched it move about. That first marble moved quite naturally on my 90 MHz, 16MB RAM PC.

By the time I finished building my world, I realized that I could share a realistic looking world of 4 moving marbles, 40 fixed objects, and 5 moving objects, all with their own embedded physics, over the Internet. For me, that level of performance is quite satisfactory as a platform to begin building interesting shared worlds. I do envision some optimizations I could apply to my code in order to speed things up further. For example, my collision detection strategy compares the location of each movable object to the location of every other object without consideration to any regions on the board. For 4 marbles and 40 objects, my code makes 160 comparisons per animation loop. Probably, with some added logic to the collision detection code, I could cut those comparisons down to 20 or so per animation loop with minimal added overhead.

2. My world could be placed within any other world.

As I created my world, I kept reminding myself that my world could be contained inside of any other 3D architecture imaginable. I continue to be fascinated with the possibility of creating smaller worlds of interest that cyber-participants can carry around with them and use with others. VRML 2 is a language with no inherent unit of measure. If I had the computing power available to me, I could simulate interactions with objects by starting at the atomic scale and build right up to the scale of galaxies just by building each object out of the appropriate virtual atoms. Any VRML object made available on a Web server can be incorporated into any other VRML world. A single VRML object created by a single person and placed in a single VRML 2 file on a Web server could almost immediately appear in millions of different VRML worlds if the network passed it around and others made reference to it in their worlds. The next step is to create objects that can interact with other objects. It appears to me that Java could add behavioral logic to virtual objects until VRML finds a way to standardize the process. Then, smaller objects of interest can come alive in virtual worlds all around the planet. My world, just like any other VRML and Java based world, could literally become an overnight sensation.

3. Lots of Interesting worlds to build with this technology.

As I created my world, I kept thinking of all the other interesting worlds that I could be creating instead. Almost any existing board game could be implemented using this technology. But, I am most interested in worlds that allow participants to design new worlds or board games and try out different rules to stumble across successful designs. I believe this technology will allow groups of geographically dispersed people to design new worlds which no one individual in the group would have ever imagined on his or her own. I believe this technology could allow consumers to visualize new products and gain the support of other consumers who could then contact manufacturers about making the product. Spontaneous connectivity of groups that share virtual worlds without consideration of geographical distance is an affordance of the Web that has never existed before. It is very possible the most interesting worlds have not yet been imagined.

4. There is a lot that can be done with latency.

Although I found it not feasible to use this technology to create a Quake world or Doom world that would keep up with the latest *shoot-em up* games, I did find it to be a technology that would work well with any board game that requires contemplation and reflection before each action is taken. As long as decisions need not be made in the sub-second time frame, any simple world that involves sharing information could be developed with this technology. 3D chat worlds work fine with the latency of the Internet, but it is time to reach beyond simple chat worlds.

5. A server is more unbiased than many moderators or facilitators.

As a person who always disliked participating in games where people cheated and who sometimes disliked how long it took others to make a move, I found the use of a server as a facilitator to offer some really great benefits. A server enforces rules reliably without a chance of personal bias against any player (unless the bias is programmed in). A server obeys agreed upon time limits. And, in the extreme, a server can actually take action on behalf of a participant if need be (an interesting possibility is to finish a game for a player who leaves the world before the game is finished). A server can also create teams and connect players in a fair and unbiased manner without hurting anyone's feelings.

Outside of game playing, a server can also be programmed to enforce rules to provide fairness to a group discussion. A server can make sure each participant gets equal time in group communications.

6. There is a lot of potential for a Java 3D API.

By the time I finished using the EAI, I realized that most of the value of my world was provided by the Java instead of the VRML 2. Yet, the VRML 2 viewer provided much of the freedom I gained from using VRML 2. The VRML 2 viewer provides me with the ability to explore my world with six degrees of freedom. I certainly would not have wanted to program that functionality myself.

Yet, if there were Java classes I could use to easily create the VRML 2 viewer functionality I liked, I might not want to limit myself to VRML 2 as a file format for 3D models. I believe the approach Sun Microsystems, Inc. is taking with designing their 3D API is a valid one. With Sun Microsystems, Inc.'s 3D Java API, I will be able to load a Java based scene graph with any 3D file format I wish by using the appropriate loader class. I see much potential in a working implementation of a Java 3D API. Although others have been working on Java based APIs for sharing VRML worlds, Sun Microsystems, Inc. should be able to leverage their API through their association with other Java innovations.

7.2 Experiment Results

I ran twelve subjects through two simulations each. Each subject was placed randomly in a group with two other participants. The first simulation had a group try to reach goals in a world designed by another group. The second simulation had participants design the rules and the world before attempting the same objective in that world as well. I compare results between the two groups that interacted with the same world. The first group participated in a world the second group had designed. The second group participated in that same world with the advantage that they had already participated in a world of another group's design. Table 7.1 shows the simulation scores attained by the participant groups. The first row in the table presents the number of simulation frames the group required to finish the simulation when that world had been designed by a previous group. The second row in the table presents the number of simulation frames the group that

designed that world required to finish the simulation. The lower the score the better.

Table 7.1 Simulation Completion Times

World Iteration Number:	1	2	3	4
Play Others Design:	1441	2200*	710	1646
Play Own Design:	1116	1805	685	1178

* Estimated as group only reached 9 of 10 goals in maximum time of 2000

Not surprisingly, in all four cases, the group that had designed the board completed the simulation faster than the group that had no influence in its design. On average, the designers completed the simulation 20% faster than the non-designers. Yet, in all four cases, the participant groups ran the simulation where they were not designers first. They ran their own designed world second. By observing the individuals work, I noticed that they experienced a significant learning curve and I feel that their better performance in their designed world could be attributed to the fact that they were just better at performing in general since it was their second time around. Two types of learning were taking place during the simulations: First, technical learning involved understanding the interface and how the system responded to human interaction. Second, collaborative learning involved understanding how to work with others when success relied on that cooperation. I believe the participants had moved up the learning curve significantly between the first and second simulations. Yet, I also believe they still had a lot more to learn after participating in two simulations.

Figure 7.1 shows the results of the ten objective questions of the post-participation questionnaires. For each question, the left-most bar represents the first questionnaire filled out by participants immediately after playing a collaborative world of another group's design. The right-most bar represents the second questionnaire filled out by participants immediately after playing a collaborative world of their own design. The y-axis represents the average selected by participants on a scale of 1 to 7 where 1 means strongly disagree and 7 means strongly agree.

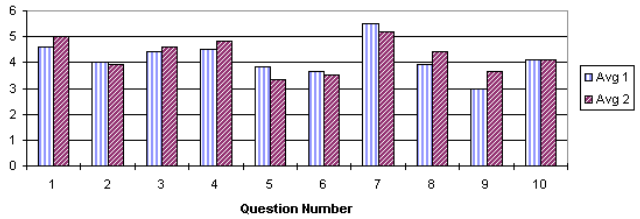

Figure 7.1 Post-Simulation Questionnaire Results

Perhaps the differences between the two questionnaires are not as significant as the average response value for each question. I now discuss each of the ten questions and the responses in detail:

Question 1: I enjoyed participating in the simulation.

	Mean	Hi	Lo
Questionnaire 1	5.58	7	4
Questionnaire 2	6	7	4

I am pleased to find that the participants generally enjoyed participating in the simulation. Participants enjoyed participating when they were involved in the design more than when they used another group's design. Of all 24 questionnaires completed, not a single participant answered on the disagree end of the scale. I believe their enjoyment would improve even more had they not experienced some of the technical difficulties I discuss later.

Question 2: I became immersed in the simulation with little awareness of other things going on around me.

	Mean	Hi	Lo
Questionnaire 1	5	7	2
Questionnaire 2	4.92	7	1

I found the responses on question 2 to vary greatly by individual. I am pleased with the level of immersion reported by participants, but am not sure of how they define immersion. I understand why the participants reported as sense of less immersion in the design phase. Many participants did not watch the computer screen while others were taking their turn designing. Instead, participants looked away during that time. Certainly, I could have shown the movements of design pieces of other participants while they were moving pieces about on their monitor. I thought that would be distracting. I now believe participants would have felt more immersed if they could follow other participant's actions. I found it very curious that most participants were not planning their next upcoming move more when it was not their turn. I observed participants who reacted as if they should not do anything unless it was their turn, including thinking about the task at hand.

Question 3: I would like to participate again with the same group of participants.

	Mean	Hi	Lo
Questionnaire 1	5.42	7	2
Questionnaire 2	5.58	7	3

Overall, participants were quite enthusiastic about participating again. I am pleased with the fact that participants wanted to go again even when they encountered technical difficulties. Participants were quite eager to participate with the same group even though they did not know their fellow participants in many cases. Participants were more eager to participate again after they had been involved with designing the world.

Question 4: I would like to participate again with a new group of participants.

	Mean	Hi	Lo
Questionnaire 1	5.5	7	4
Questionnaire 2	5.83	7	3

Participants were even more enthusiastic about participating with a new group of people. I observed that each participant had his or her own unique strategy that did not necessarily work well with other strategies within their group. I believe participants wanted to go again with a new group because they had high hopes the next group would think more like themselves. Again, participants wanted to participate even more after being involved with the design of the world. I also believe that they wanted to go again because they were learning some things that they needed more time to figure out completely.

Question 5: I thought the technology performed well.

	Mean	Hi	Lo
Questionnaire 1	4.83	6	4
Questionnaire 2	4.33	5	2

Participants did not rate the technology particularly high, but I am pleased with their responses given the technical problems encountered. Of all 12 simulations, two had minor problems that effected results and two had major problems. The other eight simulations ran well enough that the participants were not aware of the problems. I believe the major cause of technical problems was the fact that users used their mice as pointing devices much more aggressively than I anticipated. The CosmoPlayer browser obviously runs code whenever the user moves the mouse to track positioning. When I ran my timing studies, I did not move the mice so often and was overconfident that the worlds would stay in synch.

Twice the technology crashed and I had to start the simulations over. The first time the group had just finished the design of their board when the Java console returned an *Invalid Field* exception. I had not experienced that error in any of my testing. I believe that crash did not affect their results since I started the simulation again from that exact point after a five-minute hiatus. The second crash occurred 680 frames into the active simulation — that crash was much more frustrating for me. I had the group fill out their questionnaires at that point in time, but then restarted the active simulation to get a timing result from the group. The world ran acceptably the second time. I did not have time to investigate the reason for the crash. I did notice though that the group of the crash had the most active mouse habits.

The two simulations with minor problems could also be attributed to mouse use. In those simulations, frame rates sped up and slowed down dramatically on the fastest PC with the participant at that computer feeling a bit frustrated and annoyed. Eventually, the speed swings settled down in both cases, but not after significantly affecting the experience for that participant.

Question 6: The interface was natural to use.

	Mean	Hi	Lo
Questionnaire 1	4.67	6	2
Questionnaire 2	4.5	6	3

Participants did not rate the interface particularly high, but did a good job of indicating what would have made them rate it higher. Although I reviewed the significance of each control with them before the simulation, many told me they had forgotten which control was which. I believe they would have rated the interface better had I included labels for the controls. I felt that labels would take away from the immersive experience and had decided not to use them. Others complained about the cues from the slant queue wishing the slants were updated more often. I now believe I should update the slant queues as soon as a new slant is chosen by a user. Instead, I was waiting until the next slant change. By waiting, users were unable to tell what other participants had selected during their turn until after they took their own turn. A participant could not make an informed decision under those circumstances.

Lastly, participants found the movement of the board to jump at times. I believe that I could smooth the movement out by changing some of the VRML code, but I think some of the jumpiness is from code embedded in the VRML viewer itself. I need to spend more time investigating my options for smoothing out the board movement.

Question 7: I felt like I was treated as an equal in the simulation.

	Mean	Hi	Lo
Questionnaire 1	6.5	7	4
Questionnaire 2	6.17	7	3

The participants overall strongly agreed that they were treated equally by the server. I am pleased with this result since I suggest that a server can make collaborating much more enjoyable in certain situations since it is a just and fair facilitator of human communications. I believe the only reason participants were not unanimous in strongly agreeing with the statement was that during the rounds with technical difficulties certain participants thought their computer was misbehaving worse than others. I thought it interesting that participants perceived the difference without actually seeing anyone else's machine. The fact that participants rated fairness lower during design rounds supports my belief. Both serious technical problem rounds were design rounds.

Question 8: I learned something about collaboration during the simulation.

	Mean	Hi	Lo
Questionnaire 1	4.92	7	2
Questionnaire 2	5.42	7	3

Overall, participants agreed that they learned something about collaboration by participating in the world. I learned a lot about collaboration just by watching them try to collaborate and so was hoping for even higher results. I think they would have learned more had they kept on working on additional boards and running additional simulations. Participants learned more about collaboration when they collaboratively designed the world. I noticed many individuals who had great strategies not get the response from others they needed to get their strategy fully enacted. Other participants realized what could have been done later on had they cooperated better. As I mentioned earlier, I was surprised they did not spend more time thinking when it was not their turn because they could have figured out more during the design phase.

Question 9: I had more control of what took place than I anticipated.

	Mean	Hi	Lo
Questionnaire 1	4	6	1
Questionnaire 2	4.67	7	3

The results to question 9 are difficult to analyze. Participant answers varied greatly on the scale. I was expecting the results to be much lower than they were since I did not think participants felt they had much control. Their subjective answers report their frustrations. Perhaps they did not anticipate having much control in the first place. Participants found that they had more control when they controlled the design of the board.

Question 10: I would like to have more control of what happens in the simulation.

	Mean	Hi	Lo
Questionnaire 1	5.08	7	3
Questionnaire 2	5.08	7	1

Participants generally wanted more control, but many reported that they learned more about collaboration by not having a lot of control. I find it interesting that participants did not rate this question higher on questionnaire 1 than questionnaire 2 since on questionnaire 1 many mentioned how they wanted the control over the design that they got in the second simulation. Perhaps the control over the design did not make the difference they had hoped for.

The following is a review of the comments from the four subjective questions asked on the questionnaire:

Question 11: Please describe your frame of mind when you started the simulation.

First Questionnaire:

1. A little tired, a little excited
2. Enthusiastic
3. Tired
4. Confused but anticipating
5. Curious
6. Tired from writing papers but otherwise in good spirits
7. I was confused at first because of the numerous operations available to me
8. Curious
9. The demonstration helped clear some of the cloudiness I had after reading the introduction so I was much better prepared, but still a little less on the what to do or the purpose.
10. A little tense and wanting to do well.
11. Open to whatever was going to happen.
12. Interested in whatever would occur.

Second Questionnaire:

1. Looking forward to trying
2. Thought ahead. Was excited about the game.
3. Tired
4. A little tired, a little excited
5. Alert
6. Planning
7. Hopeful about being able to create a good board
8. Anticipation of playing my game
9. Anticipating
10. Curious again
11. I was uncertain about where to place the obstacles at first.
12. Having already done it once, I was more aware of the consequences/results. So I was more comfortable.

I found the comments before the design simulation to be more specific to the task at hand than the first simulation. Most individuals appeared to be open-minded about the upcoming simulation instead of holding any negative expectations.

Question 12: Please describe your frame of mind during the simulation.

First Questionnaire

1. Fairly engaged
2. Discovery
3. Tense, trying to remember rules
4. Frustration, anticipation
5. A bit frustrated by the lag, but still having fun
6. More actively engaged in activity because I saw evidence of my direct manipulation of objects.
7. I became more focused as the game moved along.
8. Adapting to the delayed response
9. The slowness of the computer confused me a bit because the simulation crawled.
10. Frustrated with the lag and my catching on rather late about how to plan my movements well.
11. Concentrated on the task at hand trying to set up the next shot
12. Analyzing - trying to find out how best to play and what was going on

Second Questionnaire:

1. Intense, so much to think about
2. Was trying to find the best ways to reach the goals
3. Busy
4. Patient, a little confused
5. Absorbed
6. Extrapolating

7. Heightened awareness and focus on what was happening
8. Eager to sink the balls to prove our design was good
9. Go get 'em!
10. Idling to some extent
11. I had a better idea about where I wanted to place the obstacles as time went along
12. The smoother frame rate made it easier to comprehend what was happening, but the screen movements were far from consistent so I was still a little confused.

I found the participants to be quite engaged with the simulation based on comments about their thoughts during the simulation. I did not notice any strong differences between the first and second simulations, as both sets of comments seem to reflect similar thoughts.

Question 13: Please describe your frame of mind now (after the simulation).

First Questionnaire:

1. A little tired, a little happy
2. Enthusiastic
3. Relaxed, more alert
4. Ready to go again
5. Interested in constructing my own board
6. Interested in moving on to a game that I design
7. I now realize what was taking place. I now feel more comfortable.
8. About the same as before
9. Though I was a bit confused, during the simulation I comprehended
10. Happy - I am on a date
11. Want to do it again
12. Curious about whether we could learn more and do better next time

Second Questionnaire:

1. Still intense
2. Want to play again, want to be more effective, more goal oriented
3. Tired but alert
4. A little tired, a little bemused
5. Want to do over
6. Contemplative
7. Cool experiment. Fun. Glad to see what a difference my choices made.
8. Intrigued about the possibilities of this technology
9. Job well done, at least on my computer
10. Same as during
11. Wish I would have done a better job of placing my obstacles
12. I can see how this simulation can be useful for future applications. Call me when 4.0 comes out (hey, its a popular version number).

I found the comments about post-simulation thoughts to be quite promising as most comments reflect a state of positive reflection about the simulation. I notice no significant difference between participant's thoughts after design simulations and non-design simulations.

Question 14: Please list any frustrations you experienced while participating in the simulation.

First Questionnaire:

1. Another participant was yelling.
2. Others choosing too rapidly
3. Had trouble remembering what tools on palette did
4. Too slow, not enough obvious control over things (but I imagine that is part of the game)
5. Effects seem to be of limited use when the lag time before their start and the amount of time they last are unknown.
6. Delay in seeing future slopes (couldn't tell if my team mate chose the slope or if the computer did). The cues for what was next came up too slowly so I had to pick a slope based on choices before last.
7. The effects didn't work as well as I would have liked. They did not respond as much as I hoped.
8. Slow
9. Just the slow server
10. See above (frustrated with the lag and my catching on rather late about how to plan my movements well).
11. None I can think of
12. I wasn't really frustrated. I enjoyed learning.

Second Questionnaire:

1. Other people not doing what I expected
2. I felt my turn appeared for only short periods of time.
3. None
4. Group answers were unclear
5. Remembering what tools did what
6. Refresh rate of future slants
7. Didn't know we completed goal - one computer had the feedback.
8. During the rules voting phase, the answers returned did not seem to be in the same format as the questions. For example, I chose *3* viscosity and then saw the selected coefficient was .45 or something -- what is the relationship of scale?
9. My computer finished before others - not quite as fully integrated as it needed to be but great fun.
10. Again, it was slow
11. I was frustrated that my group apparently used most of the same types of obstacles instead of using some variation.
12. As before, the screen wasn't always consistent. This made it harder to make/anticipate moves - both mine and those of other players.

I will address all frustrations from question 14 here as I think the comments made by participants were especially insightful. As for comment 1 of questionnaire 1, I would expect participants not to hear another participant's yelling if they were dispersed all over the globe. In response to comment 2 of questionnaire 1, I would expect participants not to choose too quickly once they learned how to successfully collaborate in the world. Choosing too quickly is not an effective strategy just as choosing too quickly in chess is not a good strategy for winning chess matches. As for comment 3 of questionnaire 1 and comment 5 of questionnaire 2, I agree that I should strongly consider adding labels to palette objects initially during the learning phase. In response to comments 4, 8, and 9 of questionnaire 1 and comment 10 of questionnaire 2, I believe there is a place on the Web for slower, reflective activities. Marbles world is not meant to be a typical

video game experience. In response to comments 5 and 7 of questionnaire 1, I think I should consider helping out first time participants by giving them more information about effects until they understand how to ascertain the timings themselves. In response to comment 6 on both questionnaires, I agree whole heartedly that I should update the slant queue immediately instead of waiting for the next slant change. In response to comment 1 on questionnaire 2, I believe participants' actions would come together more after participating together for a period of time. My whole objective centers around the belief that my world would teach better collaborative behaviors. In response to comments 7, 9 and 12 of questionnaire 2, I must make the technology work acceptably each time as technical problems effected the opinions of participants greatly, although I believe most expected problems at some level and that helped keep them enthusiastic. I believe they saw their feedback as important in the process of making a better simulation. Technical problems cause me to give up on a virtual community more than any other factor. I have no reason to believe I am different in that behavior than a typical Web citizen. In response to comment 8 of questionnaire 2, I agree that I should provide more integrated feedback to group decisions. In response to comment 11 of questionnaire 2, I believe I must provide a mix of obstacles that make for interesting combinations of strategies. I have no doubt that better obstacles could have enabled better designs by participants. Yet, I found two of the groups to have created very successful looking designs using the obstacles they were given.

7.3 What To Do Next

I have found the results from this project to be quite encouraging, yet I believe there are some steps I could take to make the next go-around even more successful. Most importantly, I need to fix the technical problems that effected 4 of the 12 simulations. The server I wrote and used for the project contains very few lines of code dedicated to the inter-world communications that keep multiple copies of the world in synch. In fact, each client reports timing data only once every 50 frames. I can experiment with more frequent time reports that show promise of correcting divergent behaviors more rapidly. I can consider stopping a client process when it gets way ahead of other clients in order to let the other clients catch up gracefully.

I also can experiment with other interface designs including using labels for controls until participants feel they no longer need them. I definitely must consider updating the slant queue feedback as soon as a slant is chosen by any participant. I had not thought through how important the upcoming slant information would be for taking the best action based on the prior participant's turn. I would like to allow a user to drag and drop a new obstacle from the obstacle palette instead of having to click first, move the mouse, and then drag. Currently, using the external authoring interface there is no easy way to create a new VRML object without clicking on something in the world first. Perhaps I can come up with a creative solution if I give the problem more thought. I also must consider providing better rules information for first time participants. Although I believe a seasoned user could figure out the rules based on interacting with the simulation, I would not want to lose potential long-term community members due to initial frustrations.

Once I fix the existing technical problems, I can open up the world to others in a way that they could help design new rules, obstacles and effects for inclusion with the simulation. I personally have 50 or more ideas I would like to implement myself, but I believe I must allow the community to come up with ideas. I feel that most of my ideas would be independently suggested by other participants in due time anyway. As computing power increases, I believe the obstacles and effects in the world could become very sophisticated. Marbles world could become significantly more active than its current implementation and the marbles might even be able to take on different shapes and behaviors. Marbles world would not have to exist in a single x-y plain but instead could include multiple boards stacked on top of each other along the z-axis. Yet, to really test my ideas and beliefs, I need to create a community around this project and let the community drive the project in whatever direction they want. Perhaps I would become a facilitator more so than a designer. A community working on a collaborative Web project needs technical assistance in determining what is possible given computing trade-offs related to resources. I have experienced the frustration of working on technical ideas without any guidance as to what is a reasonable design for a given computing platform. Many of my Lotus Notes applications were designed without any sense of whether they could perform well on a computing platform my users would be using. I need to come up with reasonable guidelines for obstacle and effect designs of community members.

In the very long-term, applications like marbles world should be able to be experienced by hundreds or thousands of participants at the same time. Server-less, distributed architectures show promise of being able to support large amounts of users. I can begin to look at alternate architectures that rely on multicasting in order to scale up marbles world to larger communities.

Yet, in the short-term, I can consider building other applications that use the architecture I used for marbles world. I believe this architecture will work well for group design and play of golf courses, croquet lawns, ski slopes, and all kinds of applications where the rules are already somewhat standard and require little new learning. To me, the interest will continue to be in building interesting communities on the Web where people meet for specific purposes that are engaging and educational. I continue to hope that technologists can develop the infrastructure with which Web participants can create the communities they want most. I hope Web surfers will be able to work together to create interesting worlds. I believe my work shows that many users would be interested in trying out such communities at least once. I found all my subjects to be very enthusiastic before, during, and after their initial two simulations. In fact, all but one requested to be invited back once I spent more time building a better application.

I have learned that building networked applications is very difficult to do by oneself. The server code turned out not to be the monster I had anticipated thanks to existing Java classes that hid the complexity. Yet, the testing certainly was difficult when working with computers that physically existed at least 20 feet from each other. I can invest more time in creating better test methods. Lastly, I must continue to consider new technologies to use when implementing 3D multi-user collaborative worlds on the Internet. The Java 3D API from Sun Microsystems, Inc. shows great promise as a delivery platform. As my creative thought process settles into more concrete paths of action, I can apply better engineering techniques to quantify and test my development efforts. Through this project, I have learned that the creative thought process can continue for weeks at a time. I need to run

more concrete tests before I can begin to weed out ideas based on merit.

ACKNOWLEDGEMENTS

I wish to acknowledge my advisory council: Dr. Michael Danchak and Don Merusi at the Rensselaer Polytechnic Institute; Dr. Tom Furness at The University of Washington, who directs the Human Interface Technology (HIT) Laboratory - a place where I was fortunate to access a lot of information, use a wide range of computing resources, and debate with a talented group of staff researchers and students.

More generally, I must acknowledge all the individuals out there in cyberspace who responded to my email inquiries requesting more information on VRML, Java APIs, and philosophical questions. I must also thank Ian and Shirley Campbell for helping with the tedious job of proof-reading into the wee hours of the morning. Lastly, I want to acknowledge specific colleagues at the HIT Lab who helped me refine my project goals and kept me going during my slower periods of progress: Dace Campbell, a virtual architect, helped me realize that I should better leave the design of public cyberspace to others and instead focus on what happens inside the architecture; Susan Tanney, also a virtual architect, kept me excited about the significance of my work; Suzanne Weghorst, a human factors specialist, provided me with other opportunities to practice the tools of the trade in VRML 2 world creation by assigning me meaningful lab projects; Mark Billinghurst, a cyberspace engineer, debated my approach to solving problems; Toni Emerson, a cybrarian, found me relevant magazine articles and books which helped me think through my ideas.

REFERENCES

[1] Mead Paper, The History of Paper, <http://www.mead.com/mead/history.html> (Accessed 28 January 1997)

[2] Blum, Daniel J. and David M. Litwack, Addison-Wesley, The E-Mail Frontier Emerging Markets and Evolving Technologies, DonMills, Ontario (1994)

[3] Community Access to Technology Assisted Learning (CATAL) , History of the Internet, <http://tdi.uregina.ca/~wetsch/internet/history.html> (Accessed 28 January 1997)

[4] Reid, Elizabeth M., Electropolis: Communication and Community On Internet Relay Chat, <http://www.ee.mu.oz.au/papers/emr/electropolis.html> (Accessed 28 January 1997)

[5] Bartle, Dr. Richard, Interactive Multi-User Computer Games, <http://www.oise.on.ca/~jnolan/muds/about_muds/mudreport> (Accessed 28 January 1997)

[6] University of Illinois Alumni News, From Plato to Iris: The force behind Lotus Notes, <http://ftp.cs.uiuc.edu/CS_INFO_SERVER/ALUMNI_INFO/newsletter/v1n2/iris.html> (Accessed 29 January 1997)

[7] Reid, Elizabeth M., Cultural Formations in Text-Based Virtual Realities, <http://www.ee.mu.oz.au/papers/emr/cult-form.html> (Accessed 29 January 1997)

[8] Simulation Interoperability Standards Organization, The DIS Vision: A Map to the Future of Distributed Simulation, <http://siso.sc.ist.ucf.edu/docref/general/vision/index.htm> (Accessed 29 January 1997)

[9] Roehl, Bernie, Distributed Virtual Reality -- An Overview, <http://sunee.uwaterloo.ca/~broehl/distrib.html> (Accessed 29 January 1997)

[10] Mandeville, Jon et al, GreenSpace: Creating a Distributed Virtual Environment for Global Applications, <http://www.hitl.washington.edu/publications/p-95-17> (Accessed 11 February 1997)

[11] Videotopia, Arcade Games, <http://www.videotopia.com/games.htm> (Accessed 11 February 1997)

[12] VRML Architecture Group, The Virtual Reality Modeling Language Specification, <http://vag.vrml.org./VRML2.0/FINAL/spec/index.html> (Accessed 31 January 1997)

[13] The World Wide Web Consortium, HyperText Markup Language (HTML) <http://www.w3.org/pub/WWW/MarkUp/> (Accessed 31 January 1997)

[14] blaxxun interactive, Blaxxun Community Platform, <http://ww3.blacksun.com/products/index.html> (Accessed 1 March 1997)

[15] OnLive! Technologies, Inc., OnLive! Product Line, <http://www.onlive.com/prod/> (Accessed 4 March 1997)

[16] Sony Corporation, What's It About, <http://sonypic.com/vs/about.html> (Accessed 1 March 1997)

[17] OZ Interactive Inc., The Future is Now, <http://www.oz.com/> (Accessed 12 March 1997)

[18] Mandeville, Jon et al, GreenSpace: Creating a Distributed Virtual Environment for Global Applications, <http://www.hitl.washington.edu/publications/p-95-17> (Accessed 11 February 1997)

[19] Sun Microsystems, Inc., Software Solutions: Sun's Java Products <http://www.sun.com/java/sw.html> (Accessed 14 February 1997)

[20] Living Worlds, Living Worlds: Making VRML 2.0 Applications Interpersonal and Interoperable, <http://www.livingworlds.com/draft_1/index.htm> (Accessed 13 February 1997)

[21] The Virtual Reality Modeling Language Appendix C. Java Scripting <http://vrml.sgi.com/moving-worlds/spec/part1/java.html> (Accessed 9 February 1997)

[22] Yergin, Daniel, Simon & Schuster, New York, NY, The Prize, (1992)

[23] The VRML Consortium, Members List, <http://vag.vrml.org/consort/Members.html> (Accessed 19 February 1997)