Web-based Collaborative Design Environment

by

Hai Ning

Bachelor of Architecture
Tsinghua University, 1997

Bachelor of Engineering in Computer Science
Tsinghua University, 1997

Submitted to the Department of Architecture in Partial
Fulfillment of the Requirements for the Degree of
Master of Science in Architecture Studies

at the
Massachusetts Institute of Technology

June 1999

© 1999 Hai Ning. All rights reserved.
The author hereby grants to MIT permission to reproduce and publicly distribute,
including via electronic copies, all or any of this thesis.

Signature of Author ____________________________

Hai Ning

Department of Architecture
May 1999

Certified by ____________________________

William J. Mitchell
Professor of Architecture and Media Arts and Science
Thesis Advisor

Accepted by ____________________________

Roy Strickland
Associate Professor of Architecture
Chairman, Departmental Committee on Graduate Students
Thesis Readers:

William L. Porter
Norman B. and Muriel Leventhal Professor of Architecture and Planning
Department of Architecture and Planning, Massachusetts Institute of Technology

John R. Williams
Associate Professor of Civil and Environmental Engineering
Department of Civil and Environment Engineering,
Massachusetts Institute of Technology
Web-based Collaborative Design Environment

by

Hai Ning

Submitted in the Department of Architecture on May 1999 in Partial Fulfillment of the Requirements for the Degree of Master of Science in Architecture Studies.

Abstract

This thesis explores the collaborative design process facing the challenge of the new era of information technology. Architecture design is often a tremendous collaboration process participated by multi-professions. Successful communication of requirement and intent are critical factors that could determine the overall success of the project both in concept and detailed design.

In the information technology revolution, communication becomes the center of this new movement. We now possess much more powerful communication tools and methods than ever before. Computer is not just about computation anymore. It is about communication and interaction. Many CAD software packages have appeared to be very helpful in architecture drafting, documentation and even representation. However, we demand more in design thinking, in idea sharing, and in collaboration.

Thus, it is natural to bring up the issue of effective communication within the context of architecture design process. This thesis anticipates the emergence of virtual reality modeling language (VRML) as an economical alternative not just to architectural presentation as many have already demonstrated, but also as an effective tool in architectural design process to help designers express their own idea, and understand other's as well. As a major practice of this thesis project, I will describe a prototype system VirtualBlock that I have built to illustrate the possibility to allow designers interact with each other mediated by computer network easily and efficiently.

Thesis Advisor: William J. Mitchell
Title: Dean, School of Architecture and Planning,
Professor of Architecture and Media Arts and Science
Acknowledgement

Dedication

To Bing, for the years and years of love and support.

Many thanks to

Bill Mitchell, Bill Porter, John Williams, for your insight and encouragement in this adventure.

All my SMArchS colleagues, for your friendship and resourceful help.

My friends in I.E.S.L., for your inspiration and friendship making these two years special.

Last but not least, my parents Yuanzhong Ning and Zhanglin Zeng, for your faith and support.
Table of Content

Abstract

Chapter I Introduction and Background

Chapter II Computer Mediated Collaborative Architectural Design
1. Collaboration in traditional social science
2. Collaboration in Architectural Design
3. Computer Mediated Collaborative Design
4. Communication in CSCW

Chapter III Architectural Representation and Virtual Reality
1. Graphic Architectural Representations.
2. Architectural design in virtual environments.

Chapter IV Collaborative Design System Implementation
1. VRML Introduction
2. Java: an ideal tool for collaborative design system.
3. implementation of multi-user VRML world

Chapter V VirtualBlock
1. Overview
2. Related Collaborative Web Research
3. VirtualBlock System Architecture
4. Runtime Environment Issues

Chapter VI Conclusions

Bibliography
Chapter I Introduction and Background

Architectural design is a complicated process that involves input from various professionals: architects, structure engineers, mechanical engineers, urban designers, developers, interior designers, and more. Even if we narrow the scope to “pure architectural design,” which may only include shape and space design, there is a huge amount of information going back and forth to help designers coordinate the decision making process.

Since the communication of ideas is of critical importance to designers, many methods have been explored to achieve more efficient information exchange. Drawings and models are probably the oldest, and some say still the most effective, bridges to get ideas across to brains.

Some architecture schools list “fair to good” drawing skills as one of the major prerequisites for admitting new students. Many architecture schools provide drawing and sculpture courses to help students present their ideas in design studio. And almost all architecture schools assess their students’ design work based on their drawings and physical models. When architects present their work to clients, they often resort to these traditional media, such as presentation drawings and realistic models, to convey their ideas and concepts to people who may not understand, and may not have to understand, site plans or detail drawings. Again, when they consult structure engineers, architects will present another set of drawings or models with only the information necessary to provide the engineer with a clear understanding.
However, things are now changing rapidly in the world of architecture design. With the overwhelming arrival of information technology, the computer is transforming, or has already transformed in many cases, the way we do things. Communication in design, which plays such an important role in the collaborative design process, is no exception to the rule of fast-paced technological change. The new Information Technology wave is at heart a revolution in the field of communication. Everything is de-materialized, digitized, or "bit"-ized for more efficient transportation and easy sharing. Many offices have by now achieved “paperless” operation.

Architects are not sitting on the sidelines, by any means. Since the invention of Computer Graphics, technological innovators have been trying to make the computer play the traditional role of pencil and clay. In many present-day architectural practices, drawings and models are produced in digital format using Computer-aided Design (CAD) tools. Needless to say, the computer has proved invaluable in a fairly large number of daily routines carried out by architecture firms. In the field of construction documentation, where imagination and creation are dominated by exactness and precision, it’s amazing what a computer can perform when compared to human hands. Computers excel at graphic representation: they possess the ability to render a realistic image of a building,
or to carve a precise plastic model based on a computer-generated model, out of certain special materials using a "Rapid Prototype Machine."

![Figure 1-5 Realistic rendering](image1)

**Figure 1-5 Realistic rendering**

![Figure 1-6 Plastic model produced by Rapid Prototype Machine](image2)

**Figure 1-6 Plastic model produced by Rapid Prototype Machine**

However, CAD systems have been of little use, and in some cases a downright hindrance, in situations where creative thoughts need to be generated by exchanging ideas among co-workers, as in the early stage of a design project. This is somehow self-contradictory when considering that improved communication is basically what the information revolution is all about. While we have been trying our very best to exhaust the graphics ability of computers by designing various construction documentation, rendering and image processing software, it seems we have all ignored that the one great thing about computer, the dumb metal box with a heart made out of chips and wires, and can only understand language as simple as 0 and 1, is its communication ability.

Today, at least in the academic area, it is next to impossible to find anyone who has not heard of email, the World Wide Web, etc. At the height of information highway mania, people are now more interconnected than ever before. You can email your friends, host a web site about yourself, and chat online with people you do not know and probably will never meet physically. Jargon words like telepresence and videoconferencing have quickly entered the popular vocabulary.
Negroponte says, “Computing is not about the computer any more. It is about living.” I would like to add that it is also about communicating and interacting. After being digitized, in no time your ideas or thoughts could be sent to the other side of the globe in the clear form of 0s and 1s, suffering minimal or even no loss or tampering. Thus, the issue of applying this wonderful usage of computing to the field of architectural design seems inevitable. In fact, one would be surprised and also shamed that our current exploration of CAD largely remains at the level of CAP – Computer-aided Presentation, rather than helping in the actual design process, as the acronym suggests.

Under this circumstance, it is natural that the idea of maximizing the computer’s communication abilities in the context of architectural design comes to our consideration. Before we start out to design such a system, it is critical to understand the essential role played by collaboration in the whole procedure of a design project. The following Chapter II focuses on the nature of collaboration, and various collaborative design models. I will discuss the traditional architecture representation and the new Virtual Reality technology in Chapter III. Chapter IV introduces the major cutting-edge technology on which the VirtualBlock system was built – the Virtual Reality Modeling Language (VRML) and Java programming language. In Chapter V, we will look into the details of the Virtual-Block system design, its architecture, and run-time environment. The conclusion, Chapter VI, will offer some final thoughts on the assessment and possible extension of this prototype.

---

1 P6, Being Digital, Nicholas Negroponte, Alfred A. Knopf, Inc., 1992
Chapter II Computer Mediated Collaborative Architectural Design

1. Collaboration in traditional social science

Architectural professionals have been studying design activity and the design process for a long time. Most of this type of research is carried out based on the experiences of individual designers. Teamwork in the context of architectural design has been studied relatively little. However, with the development of product design evolving increasingly towards integrated activity and multi-disciplinary participation, collaboration becomes of even greater importance in the normal professional design process.

People have tried to define collaboration in many ways. Interpretations range from “just working together” to a “synergistic collaboration of like minds”. Merriam-Webster defines collaboration as “to work jointly with others or together especially in an intellectual endeavor.” Here, the keywords are “jointly” and “together,” which reflect the essence of collaboration – a group of people working together in some kind of intellectual manner to achieve a shared goal. Kvan noted, “It can be thought as joint problem solving. It means working with others to find solutions that are satisfying to all concerned. This consists of all parties agreeing on the problem’s definition, sharing concerns as valid and digging into issues to find innovative possibilities. It means being open and exploratory. It implies a deep level of trust and acceptance”¹.

Easy as it might be to define, successful collaboration is often difficult to achieve. The root of the collaboration “problem” is the fact that there’s often tension between individual and collective rationality. The famous case is the social dilemma of “tragedy of the commons” illustrated by Hardin in 1968². He described a group of herders having open access to a common parcel of land on which they could let their cows graze. It is in each herder’s interest to put as many cows as possible onto the land, even if the commons is damaged as a result. The herder receives all the benefits from the additional cows and the damage to the commons is shared by the entire group. Yet if all herders make this individually reasonable decision, the commons would be destroyed and all would suffer.

¹ Thomas Kvan. But is collaboration? ECAADE.
This is a perfect example to show that, in many cases, behavior that is reasonable and justifiable for the individual may lead to a poorer outcome for the group.

Public good is apparently the decisive factor in effective collaboration. Related research shows that if individuals cannot be excluded from the public good in teamwork, there would be motivation for everyone to "free-ride" on other people's efforts. To explore the efficiency of such teamwork, Ostrom (1990) studied a wide range of communities that had a long history of successfully producing and maintaining collective goods. She also studied a number of communities which had failed partially or completely in meeting this challenge. In comparing the communities, Ostrom found that groups which were able to organize and govern themselves are marked by the following design principles:

1. Group boundaries are clearly defined.
2. Rules governing the use of collective goods are well matched to local needs and conditions.
3. Most individuals affected by these rules can participate in modifying the rules.
4. The rights of community members to devise their own rules is respected by external authorities.
5. A system for monitoring member's behavior exists; this monitoring is undertaken by the community members themselves.
6. A graduated system of sanctions is used.
7. Community members have access to low-cost conflict resolution mechanisms.

In the context of the design field, the public good is to achieve the shared goal that everyone commits to from the outset, which is often some sort of creation of innovative artifacts. Here artifacts have a broader scope, including not only tangible objects, but also abstract theory, creative thoughts, process approaching a problem, or resolution of a problem, etc. We can regard the whole team as a small community where the goal, or in other words the public good, requires individual's effort and the rejection of the temptation of "free riding." Based upon Ostrom's analysis, it is not too difficult to organize the

---

team in such a way that everyone is clear about his (her) own responsibility and the atmosphere around the whole team is healthy. However, while such a system may suggest that there would be no “tragedy of the commons”, it does not necessarily secure a constructive and efficient way collaborative work, or shall we say the “happiness of the commons”. We have seen too many examples where even though every member in the group works industrially, and no one seems to just want to lie down and enjoy the free-ride, the whole project eventually turns out to be a disaster because of various reasons, such as lack of communication or mutual-understanding. Good collaboration alone cannot be regarded as motivating individuals trying their best to help the group. It has more to do with the skill and strategy to orchestra the action of group members in such a way that the productivity as a whole entity is maximized. In short, make one plus one greater than two.

Most design projects encountered today cannot be accomplished by a single professional working alone. What’s often required is the participation of multiple domains and disciplines. For example, a typical information infrastructure design project, such as building an e-commerce web site, would include specialists from software engineering, project management, database design and implementation, human-computer interface

![Figure II-1 Knowledge exploration during the design process.](image-url)
design, documentation and training, telecommunication and end-user domains. During
the design process, these specialists draw upon their past experiences within the con-
texts of the artifact, the context of the design, and the technical and scientific knowledge
they possess. The procedure of the design requires the experts to dedicate their indi-
vidual specialty on different aspect of the problem and integrate them together to find the
solution. As has been pointed out before, the expected outcome not only includes the
end product but also the creation process of the product, the technical and scientific
knowledge evolved from the process, which can enter a new cycle of the knowledge ex-
ploration, and thus be applied to new design experience.

However, it is often painstaking for the design participants to achieve a high level
of integration and collaboration. The reality is that while we do need specialists from
various disciplines, it's only natural to experience frustrations related to the co-operation
of these co-workers. The reason is simple: each of them has his (her) own unique past
experience, specialized work language, differences in work patterns, perceptions of
quality and success, organizational priorities, etc. The shared final target, and the con-
straints of existing technology, may lead to challenges from one against the others' con-
tribution. This “contested collaboration” can cause conflict and has a negative impact on
the quality of the design process and design outcomes.

While it's suggested that conflict is inevitable, it can also have a positive side.
Syer addressed this in his article How Teamwork Works: “Conflict is not so much some-
thing to be resolved as an experience to be explored. Conflicting views on direction and
change within a team never totally unrelated and have great value when considered as
different parts of one story. Most exercise in conflict resolution aim at compromise, yet
real difficulties arise if conflict cannot be expressed. Avoidance of conflict either drains
interest, enthusiasm and eventually trust from the team experience or results in con-
cealed tension, political infighting and the impaired performance of certain relationships
within the team. Far from diminishing, resistance will then increase.” Conflicts often
force team members to work out a compromised plan that benefits all. The end product
may not be the best solution for each individual, but it can be the best for the group as

---

5 Syer, J and C. Connolly, How Teamwork Works: The Dynamics of Effective Team Development, London,
an integral entity. Also, the emergence of conflict enforces the mutual understanding of group members, which will facilitate future co-operation.

2. Collaboration in Architectural Design

To address the collaboration issue in the architectural design process, it’s necessary to first clarify what architectural design is. There are many interpretations of architecture and architectural design. There also have been numerous studies concerning architectural design methods since the 1960s. Nonetheless, the meaning of architectural design still remains elusive. Most of the researchers tried to reflect their theoretical concern on the development of design conceptions and thus studied design methods from their academic point of view. However, architectural design as an exercise built upon the tools and methods used in architectural practices, i.e. construction of a building, has been largely overlooked. Consequently, these theoretical interpretations have been only partial in developing our understanding of architectural design.

Roderick Lawrence⁶ suggested that architectural design can be interpreted as a tripartite concept that includes:

1) The ordering of the built environment, both spatially and temporally
2) A process of decision making based on human communication, negotiation and simulation in order to define a shared goal
3) The scheme of activities, actions and plans to achieve defined goals

From this tripartite analysis, it is easy to see that, except for the common knowledge that architectural design deals with regarding aesthetical innovation and technical implementation, there is more about the process which is explicitly “political,” if by politics we mean the intentions and activities of two or more parties to define and attain defined goals. In other words, Lawrence is describing the art of collaboration. Charles Moore criticized the isolated genius depicted by Ayn Rand in The Fountainhead: "Rejecting any sorts of attitudes of secrecy or doing work in isolation is important. And

---

speaking out against the attitudes in *The Fountainhead* every chance one gets is im-
portant."7

Architecture is an activity to create innovative artifacts, which often requires the 
exploration and integration of dynamic and diverse knowledge from multiple domains, 
disciplines and contexts among specialists. Collaborative architectural design has been 
studied on and off, at least since the 1960s. The architectural design collaboration in-
volves two dimensions. The vertical one is the so-called participatory design where de-

---

sign professionals work with end-users on the project, as oppose to the horizontal one which consists of the co-operation between designers from different disciplines. In this paper, I will address the issue of the latter, interdisciplinary type of collaboration, in particular I will focus on the co-operation between architects, basically including function consideration, shape and space manipulation, material selection and so forth.

For most architects, their collaboration experiences start from group projects in the design studio at the school where they acquire their architectural education. Within the context of architectural design, the consensus is that effective collaboration plays a central role in a successful design practice. However, collaboration is often ignored in traditional architectural education. To most architects, it is only after they've started their career practice when they learn that co-operation and negotiation sometimes may not guarantee success, but lacking them will almost certainly determine failure. "The social art of design is significant in architectural practice, yet it is so poorly understood that it is hardly considered in architectural education. Practitioners gather the necessary skills only after years of experience in vitally important design negotiations."  

We denote collaboration in a design task as the activity where more than one person works on a single design problem. When a successful collaboration project is expected, four critical elements must be shared appropriately by the team members: design task, communication, representation and documentation.

- **Design tasks.** The consensus of the "public good" of the operation has to be committed to by each member before anything can be done.

- **Communication.** Asynchronous and synchronous communication will be adopted simultaneously as the project goes on. This is the most significant factor of the whole process and will be discussed in detail in following text.

- **Representation.** Drawings and models are the oldest and most popular media to convey designer's ideas in a traditional design studio. With the development of CAAD, more and more intention is drawn upon the electronic media. VRML as one of them will be discussed in Chapter VI.

---

8 Dana Cuff, 1989, 189
• **Documentation.** While shared representation determines the way in which information is presented, shared documentation refers to how the information gets organized and stored.

These are briefly the principles that stand behind the collaborative architectural design process. However, as we mentioned earlier, regulations and rules do not guarantee a healthy collaborative atmosphere. Design professionals are humans, not machines. Good design does not arise from quiescent collaboration, nor is it a serial exchange of ideas in which layers are sequentially added by participants. Analysis of successful design outcomes suggests that successful collaboration thrive on “warm, almost familiar relations among the actors, as well as conflict and, at times, tension.”(Cuff, p234). When it works, as is pointed out by Cuff, it’s due to “a team-like sensibility [which] bonded the central players who struggled together to create the excellent outcome, but these individuals did not necessarily participate equally or collaboratively. Instead, key participants played key roles; their talent and authority was reported to be essential to the building’s success.”

In 1989, the American Institute of Architects organized a series of roundtable discussions, workshops, and conferences on the subject of “excellence in design”. Although it is futile to attempt to define what excellence is and how to achieve it, there is general agreement that certain apparent but easily ignored conditions are critical in achieving excellence.

Generally speaking, excellence is achieved when the designer knows the participants and the problem well, and when this leads to a shared definition of the problem with the other participants. Other than the four elements listed above, it is also essential for design participants to work both individually and collectively in understanding the issues and exploring solutions. In the “pre-design” phase, spending substantial time on the processes of exploration and gaining the mutual trust of those involved in the project proves to be critical in determining the success of the collaboration.

---

3. Computer Mediated Collaborative Design

While much computer support for design has been developed to aid the individual designer, the industrial reality is that designers work in teams. With the development of the Internet, which brought revolutionary change to conventional information communication and sharing, there has been greater interest in using computer network as a powerful constructive environment for collaborative design.

The term computer supported co-operation work (CSCW) denotes the recently established field of study concerned with the development of systems that support multiple individuals working together with computer systems. The technologies, concerns and ideas of CSCW are not new, and have been around for many years. Rather, the emergence of CSCW as a subject of study reflects a shift in viewpoint that combines an understanding of the way people work in groups with enabling technologies of computer networking and the associated hardware, software, services and techniques.\(^\text{10}\) The field is developing quickly and has seen a series of dedicated conferences both in the United States and Europe. The idea here is that the wide-scale introduction of personal computing was followed closely by the trend to network these systems together, allowing users access to a variety of services. Finally, the information that can be transferred and shared by users has been enriched by improved display and multi-media technology, making information technology media a much more powerful competitor against traditional media.

Although it’s become widely accepted that information technology will be integrated into traditional design, and will greatly facilitate the design process, in real architectural practice very little response has been heard -- probably because of both the still limitation of technological bottlenecks and the conservative attitude from zealous tradition defenders who claim that “only hand-made drawings can be considered as architectural drawing”. However, most architectural schools are now undertaking a great move towards the Virtual Design Studio to experiment with the impact of new technology and to experience the fundamental transformation it makes to design.

A Virtual Design Studio is distributed across space and time, with information being represented electronically. And what’s more, sharing of information is made possible across space and time by computer-mediation and computer-support. The networked VDS allows geographically dispersed designers to generate, communicate, and implement design ideas through their desktop computers. The physical location of designers becomes irrelevant to the design process. They can work interactively as well as non-interactively, synchronously as well as asynchronously. The technology utilized in a VDS system includes World Wide Web, CAAD, Email, Video Conference, etc. One of the earliest experiments includes the two-week Shanghai Urban Housing Redevelopment Project carried out by six schools of architecture (Barcelona, British Columbia, Cornell, Harvard, Hong Kong, MIT, and Washington in St. Louis) in five different time zones and three continents joined together in 1994. As of today, if we take MIT as an example, there are at least one or two VDS offered by the architecture department each school year.

It is helpful to differentiate the two layers lying behind the notion of “Computer Mediated Collaborative Design”. One would be the decades-old “Computer-aided Design,” while the other one, “Computer-mediated Collaboration” is a relatively new concept.

From the perspective of design participants and the software engineers who create the computer-aided design software, the result of efforts in the CAAD field is somewhat disappointing. It is frustrating to admit that the majority of architectural practice simply regards CAAD tools as an automation of manual routine tasks, or as a more precise alternative to the tedious construction documentation process which requires nothing but patience and exactness. Some even refer to CAAD, somewhat derisively, as Computer-Aided Architectural Drafting. For instance, the most used CAAD package AutoCAD® from AutoDesk forces you to think in terms of geometric objects such as point, line, face and so on, rather than letting you work in a more advanced manner to think about wall, window or roof. Instead of making two walls meet in the corner, you “EXTEND” one line and “TRIM” it from the other one; instead of making an opening in the façade, you “SUBTRACT” a box from the solid entity representing the exterior wall. Of course you can write a “macro” to automate this process and thus be able to create
tens of even hundreds of windows in a snap. However, this temptation to over-simplify the design, and being forced to think about design in a much lower level, could damage the creative process and obstruct the flow of a designer’s thoughts. This is not because AutoCAD is not a good system, but more that the assumptions made during the design of the software about the ways of working favor particular kinds of operation.

While we have to admit that computers do show superior ability in most tiring repetitive tasks, they have the potential to be trained to be smarter and more creative design assistants. Not long ago, thanks for object-oriented programming theory, object-oriented CAAD package appeared on the stage. Object-oriented Programming has attracted a lot of attention since it came into being. Traditional programming is organized into a collection of functions, (also known as procedures or subroutines), which will be executed in a defined order to produce the desired result. By contrast, object-oriented programming focuses on the organization of software into a collection of components, called objects, that group together

a. Related items of data, known as properties.

b. Operations that are to be performed on the data, which are known as methods.
In other words, an object is a model of a real world concept, which possesses both state and behavior. It does not take a computer guru to quickly realize that this is a perfect tool to develop CAAD software, for what traditional CAAD package lacks is exactly what object-oriented software solidly possesses – the ability to abstract objects and their behavior from meaningless geometry. Therefore, a number of object-oriented CAAD software came out as the result, among which including ArchiCAD, MiniCAD, MicroStation and so on.

![ArchiCAD Interface](image)

Figure II-4 ArchiCAD interface

As I mentioned before, this is a two-fold story. However, the other side is not as encouraging as the one discussed above. Although remote communication and cooperation have been explored very extensively in those VDS experiments, a lot of questions still remain unresolved. When we consider building up a computer system to facilitate the collaboration across space and time, among issues like performance, behaviors, specifications and implementations, the most fundamental obstacles appear to be encountered on the issues of interaction and communication.
4. Communication in CSCW

Two layers of interaction seem to occur here: the interaction between participants and computer system, and the interaction between participants. What is the interaction to be provided in terms of both situations? What type of communication can afford which interaction? Not only do the participants need to share data but, how are they going to communicate with one another? One of the hot topics in recent study of the CSCW field is video conferencing, which can be taken as a typical example. How important is it? How does it contribute to the interaction between participants? And how wide does the bandwidth have to be to support it? These issues are raised when considering other aspects of collaborative systems – data exchange; participant interactions with digital models; audio links; among others.
Communication is a key aspect of knowledge exploration and collaboration. The definition of communication used in this research is “human behavior that facilitates the sharing of meaning and which takes place in a particular social context.”\(^{11}\) In a typical CSCW process, basically two types of communication occur: synchronous communication and asynchronous communication. Each has distinctive properties that are worth discussing respectively.

In synchronous collaboration, the analog is often drawn between computer-mediated collaboration and conventional face-to-face collaboration because these two types of collaboration share many features. The term “in the same space/time” which can be applied in both cases, implies that team members are working simultaneously, are generally working on the same thing, are in continuous, real-time communication and are aware of each other’s existence and activities. However, in the virtual design environment, participants do not have to be in the same physical location. Instead, they share a computer workspace in which the work of entire group is presented to each team member with continuous, real-time update. Typically, such systems are developed using the “what you see is what I see” concept, known by the acronym “WYSIWIS”\(^{12}\) where each group member sees the same view of the data on their local workstation. However, in the design practice, “WYSIWIS” can not always fulfill all requirements. A large construction would need to be presented in various ways, or through various filters for the specialty in different discipline to observe. Structure engineer wants to see the height of the truss and would obviously not be interested in the post-modern style of the door trim. Architects would only care about the comfort of the space created by surrounding walls and show no curiosity on issues like how to wire the power cable for the lamp on the wall. While a central monster database containing all the information of the project would definitely help to preserve the consistency of the documentation, it is absolutely indispensable to create different views for different group members favoring their own interest. Even in the early phase of the design, when architects co-operate to work out the basic geometric shape and space of the building, it would be wonderful if the system could provide different scales and directions of the view of the same model

---


\(^{12}\) Stefik, M, Bowbrow, D G, Foster, G, Lanning, S and Tatar D
and allow every participating architect to observe at their own preference. This would maximize the democratic participation and reduces the risk of misleading individual opinion, which is one of essential values of collaborative design.

The technologies used to support synchronous communication include video conference software, shared electronic whiteboards and specific groupware. For real-time on-line co-operation, a large amount of raw information has to be transmitted back and forth between participants in real-time. Currently, this mode of communication is limited by the need for high bandwidth networks and a common platform at each of the nodes of the distributed workplace.

Figure II-6 Asynchronous Communication and Synchronous Communication in Design process
Asynchronous collaboration on the other hand, suggests that designers may work at different times, often on different parts of the design and do not require the simultaneous presence of all team members. As a consequence, they do not need to be connected to the network simultaneously or continuously. This mode does not impose critical bandwidth requirements. A central data locker is still needed here for participants to check out and check in data they have been working on and keep the data updated and consistent.

We have discussed conflicts and tensions in collaborative design previously. It is easy for attentions to be caught when conflict is bursting out in the middle of a synchronous communication, and each party could start right there to work out an acceptable plan for everyone. In asynchronous communication, when one designer finds out something he (she) just cannot agree on the documents checked out from the central database, he (she) is basically on his (her) own at that very moment. This designer would either have to call upon attentions of other co-workers which in some extent, breaks the asynchronous communication code, and discuss it in real time, or try his (her) very best to guess why the previous decision got made in such a way. And after that, he (she) can decide to follow the same idea or challenge it with something in his (her) mind. Thus, it would be extremely helpful if the system could give a clue on the design process to help the poor guy to grasp the idea behind the current status of the data. To achieve this level of mutual understanding, it requires the system to have the ability to record designer’s ideas along the design process in an appropriate format (text, audio, video…) and store that information with the final product. When another participant comes to check out the previous work, he (she) can also check out the previous design process to obtain a much clearer understanding before continuing. Many believe that to help all parities to understand the conflict is the first and the most important step in solving the disagreement. That makes this system feature even more valuable.

Up to this point, there is no widely accepted integrated software package to support asynchronous collaboration. Although some software engineers are trying hard to come up with an integrated system, most of the experiments like VDS are carried out with the help of a variety of IT solutions combined together. Among them there are E-
mail (including multimedia mailers), FTP, DBMS, WWW etc, VRML, which has much lower requirements of bandwidth than the synchronous communication.
Chapter III Architectural Representation and Virtual Reality

1. Graphic Architectural Representations.

Though architecture did not receive widespread recognition as a liberal profession until 19th Century, graphic representations can be dated back to the time when murals were painted by our cave-dwelling ancestors. Rendered drawings, sketches, and small-scale models have traditionally been used for the representation and communication of architectural design projects. Even today, as we are entering the information era and “being digital”, drawings and models still remain the essential methods of expressing an architect’s ideas. They continue to play a major, if not entirely decisive, role in the creation and development of architectural ideas. They have provided a way for architects to explore their concepts about built form, free from the constraints of construction.

Two-dimensional architectural representation

Drawings have been criticized and defended as an appropriate way to simulate design projects. Generally speaking, architectural drawing includes a variety of representation media. Pattern books, sketches, rendered presentation drawings, construction documentation, cognitive maps, just to name a few. By looking at the common feature behind their different usage in different contexts, it’s not too hard to figure out their inherent characteristics: static, two-dimensional simulations generally at a small-scale with respect to a real-life situation. Analysis of this inherent characteristic enables us to identify the strengths and weakness of these representations as architectural design tools.

From the experience of architects, or any design professional, drawings are not just tools to record their thought, but are also a way to stimulate the creative process. Perhaps that is the reason that drawing, through thousands of years, has been the most basic skill for architectural idea expression and exploration. Most architectural schools provide drawing training courses to help new architecture students speak this new, graphic language fluently in order to communicate their ideas efficiently later on in their

---

studio work. In a broader sense, if we look at drawing as graphical symbol representation, we would have to admit that written description also belongs to this category of communication methods. In principle, symbolic communication between people is grounded on words and visual representations. Drawings have been often used to represent the final product of the design process whereas written materials are particularly convenient to document decision making throughout the process.

On the other hand, two-dimensional representation partially fails in describing the three-dimensional real world because of its intrinsic shortcomings. For non-design professionals, in this case the clients who want to understand the 2D simulation made by architects, it sometimes can be very frustrating to interpret 2D drawings and create 3D representations in their minds. The various scales of each construction details, the mysterious elevations and sections, the arcane jargon of the program specification, and most importantly, the difficulties of simulating the ambience of architecture all suggest that interpretation is not simple.

Figure III-1 Architectural sketch
Carrying with it various values and defects, two-dimensional architectural representation survives the not-so-short history of architecture, and walks along to the new high tech era. “It enables architects to control, promote or undervalue specific characteristics of design projects. It becomes a professional language that enables artistic temperament and subjective judgements to be the basis of decision, whereas interpersonal dialogue is underplayed or ignored.”

Three-dimensional architectural representation:

To avoid this common frustration, 3D representation has obvious advantages over 2D. It simulates the real world in a much more coherent way, one that can be quickly envisioned by lay people. Instead of constructing the artifact in the mind through different plan view, people get to see and feel a smaller version of the real thing. 2D representation only utilizes one of five human sensibilities, i.e. the sight, whereas 3D representation requires both sight and feel, and even sound sometimes.

Figure III-2 Small scale wooden architectural model

---

phone lines as well as faster ISDN or T1 lines, and since the computers used for viewing the files ranges from low-end PCs to top-of-the-line supercomputers.

The power of VRML becomes apparent if you compare viewing 2D images to exploring a VRML world. Suppose, for example, that you have six images of a certain area in San Francisco and a VRML file containing data describing the same general area. The images are flat rectangles showing a particular view of the city. All you can do with them is look at them. Each pixel in each image has a fixed, unchanging value. With a VRML file, however, you can view the scene from an infinite number of viewpoints. The browser (the software that displays a VRML file) has navigation tools that allow you to travel through the scene, taking as many different paths as you desire, repeating your journey, or exploring new territory according to your whim. The VRML world can also contain animated images, sounds, and movies to further enrich the experience. Sometimes, a 2D image simply doesn't convey the same amount of information as a 3D model. For example, consider the diagram shown in the right static web page (next page), which illustrates how to assemble a desk. The left VRML scene shows a 3D presentation of the same object. The added depth dimension makes it much easier to relate the illustration to the real-world desk pieces lying in the carton. What happens
mote interpersonal dialogue and decision making during the architectural design process.

Full-scale models show the capacity to overcome many of the limitations related to the interpretation of traditional architectural drawings and models, especially the overwhelming domination of the eyes. It invites people to step inside the simulation, going through the experience cycle of observing, using, modifying and re-appraising which potentially gives a more accurate justification criteria for design.

Figure III-4 A full-scale model

Of course, drawing and model making cannot be viewed as separate processes. There’s an intimate relationship between the production of a drawing and making a model. Sometimes, models are made after drawings have been produced; and sometimes drawing are produced on the basis of constructed models. Most of the time, architects work in a mixed manner. Drawings are produced to develop or elaborate design solutions suggested during model construction, and models are made to better inform architects of the consequences of associating or disassociating design ideas explored in drawings.
2. Architectural design in virtual environments.

Virtual Reality is a term that has been used in the media to describe a number of different technologies. Most people will immediately connect this to an image where people wear a head-mounted display and digital gloves, wave their arms up and down reaching out to touch virtual objects in their virtual world, or spinning the virtual steering wheel of their virtual vehicle. This is a typical example of how Hollywood distorts technology through movies like *Lawnmower Man* only to create misunderstanding among people. It is Virtual Reality for sure, but just part of it. In fact, any technology that deals with the presentation of three dimensional data in a non-linear format can be declared Virtual Reality. Sherman and Judkins\(^4\) have identified five key elements that can be used to describe the feature of a Virtual Reality system:

- **Intensive**: in Virtual Reality the user should be concentrating on multiple, vital information, to which the user will respond.

---

\(^4\) Kate McMillan, *Virtual Reality: Architecture and the Broader Community* (Sydney, Australia, The University of New South Wales, ARCH 5915 Special Research Program 2, May 1994).
• **Interactive**: in Virtual Reality, the user and the computer act reciprocally through the computer interface.

• **Immersive**: Virtual Reality should deeply involve or absorb the use. Immersion can be illustrated by Myron Krueger’s “duck test”. If someone ducks away from a “virtual stone” aimed at his or her head, even while knowing the stone is not real, then the world is believable. This is also known as “immersion”.

• **Illustrative**: Virtual Reality should offer information in a clear, descriptive and illuminating way.

• **Intuitive**: virtual information should be easily perceived. Virtual tools should be used in a “human” way.

The most popular application of Virtual Reality system nowadays is perhaps the video game. Doom, Quake, Tomb Raider, you name it. Although it seems to break into kids' life (and even adults' in some cases) not long ago, Virtual Reality has progressed from manual simulation techniques up to present computer simulations since the 1920s. Kate McMillian also constructed a timeline that traces the development of Virtual Reality.

• Late 1920s: Edwin Link worked on vehicle simulation, arguably the first forerunner of Virtual Reality technology.

• 1940s: Teleoperation technology began.

• 1954: “Cinerama” was developed using 3-sided screen.

• Early 1960s: Development of teleoperation displays using head-mounted, closed-circuit television system by Philco and Argonne National Laboratory. Morton Heilig’s ill-fated “Sensorama”.

• 1966: Flight Simulations, NASA.

• Late 1960s: Development of synthetic computer-generated displays used for virtual environments, pioneered by Ivan Sutherland.

• Mid 1970s: Krueger coined the term “Artificial Reality”.

• 1984: William Gibson published the term “cyberspace” in his novel “Neuromancer”.

• 1989: Jaron Lanier, founder of VPL research, coined the term “Virtual Reality” to encompass all of the “virtual” project e.g. “virtual worlds”, “virtual cockpits”, “virtual environments” and “virtual workstations”.

33
1990 to present: Continued research for the specific use of Virtual Reality in modeling, communication, information control, arts and entertainment.

One key advantage of Virtual Reality lies in the fact that people navigate through three-dimensional space on a daily basis. Therefore it would take no time for them to acquaint themselves in a similar computer space with minimum disorientation. Naturally, people began to think about integrating this technology into architectural design. There is actually a two-way relationship between architectural design and Virtual Reality technology.

- Architectural design can employ Virtual Reality techniques for evaluation, communication and documentation purposes. There have been a number of experiments carried out and some even put into real architectural practice. A "walk through" animation played for clients to envision the ambience of the building to be built is not unusual at the final presentation stage. A three-dimensional computer model is also commonly built and sent out through a network to co-designers for collaboration.

![Figure III-7 3D architectural model](image)

- On the other hand, Virtual Reality can employ architectural design, as one of the disciplines, which may contribute to the design of virtual environments. If you ever played video games like Doom or Quack, you would be shocked by the complicated labyrinth, and quickly realize that the only reasonable expla-
nation for such an amazing and attractive space design in the game is the
game designer being an experienced architect.

![Figure III-8 Typical scene in a VR game](image)

In this research, we're concerned with how to employ this technology to help
architectural design. As we discussed in the first section of this chapter, architectural
drawings and small-scale to full-scale models are major tools for architects to explore
ideas and manipulate design. Full-scale models, although ideal to create the built ambi-
ence, are rarely used in practice due to the difficulty of producing them, modifying them,
and transporting them for collaboration purposes.

Bearing these drawbacks in mind, if we look at Virtual Reality in the background
of architectural design, we would find that Virtual Reality has the potential of responding
to the problems raised by physical models. It is easy to construct a VR “model” with the
appropriate software package, very easy to modify after being built, and can be trans-
ferred practically at the speed of light through a computer network. What prevents Virtual
Reality from prevailing in the design field is its expensiveness in terms of both equipment
and computation. To provide the ambience of a real building up to the same level that a
physical model can provide, and to simulate the sensibility one can experience from a
real building, requires high-end computers with monster size memory and special sen-
sors which are right now not affordable to ordinary designers. The enormous amount of
data necessary to describe the senses would also need super-broad bandwidth to com-
mute back and forth. Alan Bridges and Dimitrios Charitos summarize the limitation of current Virtual Reality systems in terms of providing a realistic experience:

- Poor resolution and lack of complexity in the simulated experience
- Feedback provided for only three of the five senses
- Users do not receive enough visual, auditory or tactile kinesthetic information from their "avatars" in the virtual environments and as a result their sense of presence and their overall sense of space is limited

Nonetheless, the barriers largely consist of limitations on the technical side, and these limitations will likely be resolved in near future. The Intel's legendary co-founder and chairman emeritus Gordon Moore promulgated the famous "Moore's Law" which states that silicon-based computing speeds will double every 18 months for the foreseeable future. Based on the development of the microchip industry after his declaration, most experts, including Moore himself, expect Moore's Law to hold for at least another two decades. From this credible assumption, hardware limitations cannot impede the development of Virtual Reality in the foreseeable future. Before long, we would each have our own high-end Virtual Reality equipment set. We'll be able to design the virtual building, modify the virtual building, transport the virtual building, even actually live in the virtual building. These are not day dreams, but the future of architectural design.

---

5 Alan Bridges, Dimitrios Charitos, *On architectural design in virtual environments*, Design Studies, 18/2.
1. VRML Introduction

The Internet has given rise to an increasingly wide variety of new technologies and standards. One of the most prominent of these is HTML. Easy-to-use browsers led to the rise of the World Wide Web, and popularized the Internet, eradicating its former status as an ivory tower academic tool. At the time HTML was starting to become popular, a group of people realized that there was only so much that you could do with 2D graphics. In a meeting at the First International Conference on the World Wide Web, Mark Pesce and Tony Parisi had developed a demo program called Labyrinth that showed the use of a platform-independent graphics format. At this same conference, Tim Berners-Lee and David Ragget (the inventors of HTML and HTTP) held a discussion forum about what they termed the Virtual Reality Modeling Language, or just VRML for short.

VRML's designers wanted to create a platform-independent way to send 3D models across the Internet. To make it possible, the file format had to describe where objects were placed in 3D space and what their attributes were, such as color, texture, etc. VRML browsers would be running on everything from powerful UNIX workstations to humble desktop PCs. Silicon Graphics offered the Open Inventor file format for use, which was widely accepted. A number of changes were made to make it compatible with the Internet and WWW. This was released in May 1995. Following a number of different interpretations, a clarified version called 1.0c was issued in January of 1996.

![An example of VRML world](image.png)
In December 1995 it was proposed that the next version of VRML incorporate simple behaviors. Like everything else in the development of VRML, new pieces were being done bits at a time. VRML 1.0 described only static scenes. VRML 2.0 was to include programmable behavior but not the multi-user virtual environments of Gibson's cyberspace. They could be built on top of VRML 2.0, but multi-user virtual environments are not part of language specification. The official VRML 2.0 specification was released on August 4, 1996 — the opening day of SIGGRAPH, one of the most important conferences for the international graphics community.

The Virtual Reality Modeling Language (VRML) allows you to describe 3D objects and combine them into scenes and worlds. You can use VRML to create interactive simulations that incorporate animation, motion physics, and real-time, multi-user participation. The virtual landscapes you create can be distributed using the World Wide Web, displayed on another user's computer screen, and explored interactively by remote users. The VRML standard is defined by an advisory committee, the VRML Architecture Group (VAG), which continues to expand the language.

```
"TouchSensor" turns on a "PointLight"

#VRML V2.0 utf8
Transform {
  translation 0 0 0
  children [ 
    DEF TOUCHSENSOR TouchSensor { }, 
    Shape { 
      geometry Box { size 1 1 1 } 
    } 
    DEF LIGHT PointLight { 
      on FALSE 
    } 
  ] 
} 
ROUTE TOUCHSENSOR.isActive TO LIGHT.set_on
```

Figure IV-2 A simple VRML file

VRML provides a highly efficient format for describing simple and complex 3D objects and worlds. It needs to be efficient, since VRML files are sent through slow tele-
phone lines as well as faster ISDN or T1 lines, and since the computers used for viewing the files ranges from low-end PCs to top-of-the-line supercomputers.

The power of VRML becomes apparent if you compare viewing 2D images to exploring a VRML world. Suppose, for example, that you have six images of a certain area in San Francisco and a VRML file containing data describing the same general area. The images are flat rectangles showing a particular view of the city. All you can do with them is look at them. Each pixel in each image has a fixed, unchanging value. With a VRML file, however, you can view the scene from an infinite number of viewpoints. The browser (the software that displays a VRML file) has navigation tools that allow you to travel through the scene, taking as many different paths as you desire, repeating your journey, or exploring new territory according to your whim. The VRML world can also contain animated images, sounds, and movies to further enrich the experience. Sometimes, a 2D image simply doesn't convey the same amount of information as a 3D model. For example, consider the diagram shown in the right static web page (next page), which illustrates how to assemble a desk. The left VRML scene shows a 3D presentation of the same object. The added depth dimension makes it much easier to relate the illustration to the real-world desk pieces lying in the carton. What happens
when it is time to put the desk together? With the animation features provided by VRML 2.0, you could create an application that would allow the user to click a part shown on the screen, then watch it snap together with the adjoining pieces. If the user did not un-

Figure IV-4 Animated VRML assembling instruction versus conventional static graphical instruction

derstand what was happening, he or she could click again to separate the pieces, then repeat the process until it made sense. To see how the pieces fit together at the back, users could turn the part around and view it from the desired angle.

Whether the final goal is educational, commercial, or technical, most compelling VRML worlds have certain characteristics in common:

- A VRML world is immersive. The user enters this 3D world on the computer screen and explores it as he or she would explore part of the real world. Each person can chart a different course through this world.
- The user, not the computer, controls the experience. The local browser allows the user to explore the VRML world in any way he or she chooses. The computer does not provide a fixed set of choices or prescribe which path
must be followed, although the VRML author can make recommendations. The possibilities are unlimited.

- A VRML world is interactive. Objects in the world can respond to one another and to external events caused by the user. The user can "reach into" the scene and change elements within it.
- A VRML world blends 3D and 3D objects, animation, and multimedia effects into a single medium.

However, VRML also has its limitations. Probably the most challenging aspect of working in 3D is trying to manipulate a 3D model in 2D. Input devices like the mouse are two-dimensional, so moving them in 3D can be difficult. Even with 3D-based interfaces like Caligari's Pioneer, or the split view approach of 3D MAX, it is still difficult to keep track of exactly how things look until you get to see them in the final environment.

![Figure IV-6 Caligari's Pioneer Pro](image1)

![Figure IV-5 AutoDesk's 3D Max](image2)

Not only are computer input devices 2D, so are output devices, such as the monitor. Moving around the world helps the viewer understand the relative positions of objects, but it is still difficult to determine depth on 2D monitors. In the future, 3D monitors or head-mounted displays will become more and more affordable.

The main disadvantage to working with real-time 3D is the computing time involved. The poor little processor really has to work hard to calculate how the scene looks

---

1 Pioneer, a VRML modeling software derived from Caligari's trueSpace, a 3D modeling software.
as you move through it. The more complex the scene, the more it taxes the processor. Because of the huge calculations involved, the details of VRML worlds are purposely kept simple. That is why most VRML worlds on the WWW today can not compete with fine static 2D scenes in terms of the image quality. Complex pre-rendered animation can take days and nights to produce. Once it is finished, you can play it as fast as you want. On the other hand, VRML worlds have less than a second to compute and render the scene before your eyes detect that the motion is not smooth.

Despite the existing shortcomings mentioned above, VRML is still by far the best candidate for assisting collaborative design, if we push this technology to the limit. As a simplified Virtual Reality application, VRML has most of the advanced benefits we can obtain from a full-featured Virtual Reality system to facilitate remote collaboration: VRML is intensive, interactive, immersive, illustrative, and intuitive. It lacks one thing -- supporting multiple users. As we discussed earlier, VRML 2.0 does not provide explicit support for multiple users interacting in a single world. Handling multi-user environments is scheduled to be part of the VRML 3.0 specification. Unfortunately, in order to explore the possibility of utilizing VRML in collaborative architectural design, multi-user working in a shared space is exactly what we are looking for. Therefore, we have to resort to fairly extensive programming with the language of the network era, Java.

2. Java: an ideal tool for collaborative design system.

Java has become the hottest programming language since late 1995, when it was introduced. Sun described Java as follows:

Java: A simple, object-oriented, distributed, interpreted, robust, secure, architecture neutral, portable, high-performance, multithreaded, and dynamic language.

Sun acknowledges that this is quite a string of buzzwords, but the fact is, they aptly describe the language. An extensive discussion about the advanced features of the Java language is obviously beyond the scope of this paper. But at least some of the buzzwords boasted by Sun do look familiar to us: object-oriented, distributed, portable, and multithreaded. These words have appeared several times before in our chapters.

---

2 3D Studio MAX, a popular modeling software by AutoDesk.
discussing the characteristic of collaborative design, the advanced CAD package, the feature of VRML, etc. Finally they all come together.

Let's take a look at the benefits of developing a computer mediated collaborative design system based on the Java language.

![Object model and inheritance diagram]

Java is an object-oriented programming language. As a programmer, this means that you focus on the data in your application and methods that manipulate that data, rather than thinking strictly in terms of procedures. Suppose that we define a conceptual architectural element such as "window" as a class. Then we can add all kinds of properties to the window: color, texture, height, and width etc. Meanwhile, we define methods for manipulating the window: open, move, change color, change dimension, and etc. A sub class like gothic window, Chicago window, or international style window can also be derived from door class. They inherit all the properties and methods their super class has, but also possess their own particular properties and methods. Without doubt, object-oriented CAD will become the final product.
Java is a portable language. Because Java programs are compiled to an architecture neutral byte-code format, a Java application can run on any system, as long as that system implements the Java Virtual Machine. This is particularly important for applications distributed over the Internet or other heterogeneous networks, which is often the case of collaborative design environments. When developing a system that facilitates teamwork, design always wants the system to be able to run on each team member’s computer, no matter if it’s a Mac, PC or UNIX box. As Sun’s trademarked motto “Write Once, Run Anywhere”, “a 100% pure Java” program would eliminate the worry about system incompatibility and native codes.

Java is a distributed language, which means it provides a lot of high-level support for networking. Together with its dynamic running ability, it can be downloaded from a server and run locally on a user’s computer. This is how Web page embedded Java applet works, which makes collaboration even easier. The user does not have to carry around a computer pre-loaded with a software package like AutoCAD in order to process data. Instead, all the user needs is a networked computer. The Java program can be transferred through a network and run on a local machine. Java also provides API of
database connection. Add all these up, and a distributed collaborative design system could easily be built in a central server and be transported effortlessly across a network.

3. implementation of multi-user VRML world

Multi-user functionality is one of the vital issues for a collaborative design system. VRML 2.0, the latest release, does not explicitly support the multi-user world functionality. However, with the help of powerful Java, we can implement server-client architecture to build a multi-user environment.

A multi-user environment can be treated essentially as a kind of shared object-oriented database. The instance variables of objects in the World Model database correspond to what we would think of as "entity state" information, and updating those instance variables (using appropriate methods on the objects) is the mechanism for updating an object's state, both locally and throughout the world defined by the multi-user
technology. However, this is only half the picture. There is more to an entity in a virtual world than simply an entry in a database.

The most obvious example of this is speech. If you speak into a microphone on your system, you want your speech to emanate from your drones on all the other systems in the same virtual world. This is clearly not a database operation. If you have a video camera connected to your computer, and you use it to send images of your face that are then applied as an animated texture map onto the face of your avatar, you are again sending streaming data. From these examples, it is clear that a multi-user system has two basic aspects: a database of entities in the environment and some sort of mechanism for distributing streams of data that are emitted by those entities.

If we examine the kind of operations we typically perform in constructing, maintaining, and exploring a virtual environment, we find that many of those operations are identical to those involved in a database system. For example, if you create a new object in the environment (a table or chair, for example), you are basically adding a new record to the database that describes the world. Even if there is no “database manager” involved, it is conceptually an “add” operation.

Treating the world as a shared database also allows us to deal with issues of ownership in a very natural way. It is up to the owner of the database to decide who is permitted to do what. In addition, every object has to have a trackable owner associated with it in order to prevent virtual vandalism.

![Diagram of the World database]

Figure IV-10 the World database
The other way to view a virtual multi-user environment is to think of it as a collection of entities that are continuously emitting all kinds of data: audio, video, body positions, motion data, text, and more. To process all this streaming data, we have to break the data into frames of information, then compress each frame, and send it to the server. The server gets the streamed data, then based on a pre-programmed judgement, broadcasts it to nodes related to this information in order to update every user's view of the world.

From the preceding discussion, we can see that a number of specialized processes are needed to fully implement a multi-user system. These include:

- A database system
- A spatial partitioning system
- A filtering mechanism
- A controlling process for each entity
- One or more interaction processors
- And of course, the individual host that actually renders the virtual world for the users.

In the following chapter, we will look at a simple collaborative design system I have written for this thesis project, VirtualBlock. This multi-user system will be built around a central motion control server and a database server. The design is flexible...
enough so that it could be adapted to a multi-server topology; the host would present the same interface to their clients, but would communicate with one another using multicasting.
Chapter V VirtualBlock

1. Overview

VirtualBlock is a prototype application focusing on multi-user collaborative geometric design. It is an environment that streamlines the creation, experimentation, and testing of Web based collaborative applications. The application exploits VRML scene programming, Java applet programming, Java Database connection, and SQL programming of the Oracle database. At its core is a powerful collaboration substrate – to support synchronous multi-user applications, and a distribution substrate – which emphasizes distributed problem solving.

2. Related Collaborative Web Research

Several projects are underway to extend the capabilities of the Web to support collaborative activity: Proxy Server Collaboration, Modified Servers and CGI executables, and special purposed dedicated helper applications. Habenero is a Java based collaboration development environment. The Obliq system developed powerful migratory Web applications and systems. Brown and Najork have developed a system for implementing distributed active objects to support Web based algorithm animation. Their system uses Collaborative Active Textbooks (CAT) a system developed in Obliq to provide the necessary collaborative support needed to extend the Web. Bentley listed several areas of needed improvement in the current implementation of the HTTP protocol: client PUT, superior MIME support, improved access control, better server customization, and improved user interfaces.

Examples of collaborative virtual environments being developed include DIVE, MASSIVE, SONY Virtual Society, and NPSNET. The Swedish Institute for Computer Science (SICS) Distributed Interactive Virtual Environment (DIVE) is a fully distributed heterogeneous VR system where users navigate in 3D space, see, meet and interact with other users and applications in the environment. DIVE is a loosely coupled heterogeneous distributed system based on Unix and Internet Networking Protocols within local and wide-area networks. Consistency and concurrency control of common data is
achieved by active replication, reliable multicast protocols and distributed locking methods. DIVE supports Web access when coupled with a Browser. MASSIVE (Model, Architecture and System for Spatial Interaction in Virtual Environments) is a VR conferencing system built by Chris Greenhalgh. It is a multi-user, multi-media distributed VR system built on top of an underlying implementation of the Spatial Model of Communication. Development of MASSIVE has stopped, and work has begun on ASSIVE2 which will include more general-purpose VR functionality and will also address issues of scalability. Honda propose an architecture and the necessary protocols for supporting multi-user interactive shared 3D environments based upon the existing Web and current VRML definition. They call such an environment a Virtual Society. They list technical requirements for a Web based Virtual Society, a possible global architecture that fulfills the requirements, and a set of protocols that supports the global architecture. The protocol set includes enhanced versions of VRML and VSCP, a Virtual Society server-client communication protocol for VRML applications. The paper also reports on their initial experimental implementation of Virtual Society. NPSNET is a real-time virtual environment simulation package from the NPSNET Research Group at the Naval Postgraduate School. NPSNET currently is capable of simulating articulated humans, ground and air vehicles, and seagoing vessels in the DIS (Distributed Interactive Simulation standard) networked virtual environment. NPSNET can support about 250-300 players. NPSNET is capable of playing across the multicast backbone (MBONE) of the Internet.

3. VirtualBlock System Architecture

3.1 Overview

The VirtualBlock project is designed to simulate the process in which architects construct models with building blocks. The focus of this project is on the collaborative design process features, rather than a full-fledged 3D CAD package. For the sake of simplifying the construction and concentrating on the communication, it only provides minimum operations to place and move the blocks. However, since this type of operation has been explored very extensively in other conventional CAD software, it would not be difficult to add those functions into VirtualBlock.
3.2 Static Architecture

The static architecture of VirtualBlock is separated into three broad layers with functional interdependency among them.

- **The Application Support Layer** provides methods and operations for users to interface with the system. It exploits the Java AWT object library to implement a Java applet with a standard windows graphic user interface.

- **The Control Server Layer** explores the Java server socket classes. It actually consists of four different servers: VRML action server, chat server, drawing board server, and a database connection request server.

- **The Database Server Layer** uses the most popular enterprise database server – Oracle 7 for Windows NT. It records the design product as well as the design process.
- **The Communication Layer** consists of two sub-layers: communication between the application support layer and the control server layer, and communication between the control server layer and the database server layer.

![System Architecture Diagram](image)

**Figure V-2 System architecture**

### 3.3.1 VirtualBlock Implementation

The VirtualBlock project is written in pure Java language. The initial scene of the working space is written in VRML. The database initial part contains some SQL programming segments. At runtime, the Java applet feeds VRML browser with VRML string to construct VRML model. Meanwhile, it also sends SQL statement to manipulate the Oracle database.
3.3.2 Application Support

The application support layer requires minimal client-side environment setup. All they need is a popular Web browser (Netscape or Internet Explorer) with a VRML plug-in, which in this case, we adopt Cosmo Player 2.1 from SGI. A user from anywhere in the world, sitting on a computer of any platform (PC, Mac, or UNIX) can launch the Web browser. Through a simple HTTP connection, by merely pointing their browser to the given URL, they would be able to download the Java applet class file. The class file would then be translated by the local Java Virtual Machine to local code that can be understood by the user’s computer. Thus, a standard Graphics User Interface (GUI) would be constructed in the user’s browser window. The initial scene VRML code would also be transferred through HTTP protocol and downloaded to the user’s computer. With the help of the VRML plug-in installed in the local browser, the VRML world will appear in the user’s browser window. By clicking a few buttons, the user can construct VRML code or SQL code to interact with VRML scene and the Oracle database. A chat box is also provided for users to discuss design issues in a word medium. To further facilitate the
design, a shared drawing board is embedded in the applet, which provides another method of idea communication.

User Interface

![User Interface Diagram]

Figure V-4 Application front structure

3.3.3 Control Server

The control server layer is the heart of VirtualBlock. Written in Java, it exploits the

![Control Server Diagram]

Figure V-5 Control server structure
Server Socket technique. There are actually three servers contained in the control server layer – the VRML server, the chat server and the drawing board server. When the server starts, each server creates a socket to listen to incoming messages. Every movement that the user triggers from the front application layer that would need to update everyone’s VRML scene, chat box, or drawing board will be coded into message and sent back to the respective server socket. The server socket intercepts this message, decodes it, and does what the message requires. Most of the time, it will broadcast the message to every client that is connecting to the server to update the display of each one of them. When the message calls for database connection, it will connect to database server, retrieve the necessary information, and then broadcast back to clients.

3.3.4 Database Server

The database server VirtualBlock uses is the most popular enterprise SQL database server – Oracle 7 for Windows NT. When users interactively design the model through collaboration, every move of the block, every word that they write in the chat box and every stroke that they make on the drawing board is recorded in different arrays and kept in the memory of the control server layer. When the design achieves a certain level of satisfaction, the design would click on the save button. That triggers the control server to dump the contents of those arrays into the Oracle database through SQL statements. The reversed process happens when a user clicks to retrieve a previously recorded design. The controls server layer sends SQL statement to the Oracle database and re-
trieves data out, then rebuilds the relative arrays and broadcasts to each client, thus re-
playing the process of the previous design. The relational database contains 5 tables: scene table, block table, block movement table, conversation table, and drawing table. Each table consists of respective data necessary to record the design process. Through the relations of primary keys and foreign keys of each table, it is easy to identify the action the designer performs for each model, each block, and even each movement of the block.

3.3.5 Communication

The communication layer also consists of two sub-layers – the communication A between the control server and the front application layer, and the communication B between the control server and the database server. TCP/IP takes charge in this process. When a client downloads the Java Applet and runs it on his (or her) own computer, the Applet creates three client sockets: a VRML socket, a chat socket, and a drawing board socket. Each socket will talk to the respective server socket in the control server.

Figure V-7 Communication layer structure
layer through TCP/IP protocol. For example, user A clicks a button to generate a new block. The client Java Applet sends the information of the block that is going to be created to the server VRML socket through the TCP/IP protocol. The control server receives the message, examines it, and then broadcasts it out to all clients. The client socket in user B's computer hears the announced message, analyses it, and then translates it through the VRML External Authoring Interface to the VRML generating library. Finally a new VRML block will appear to B's screen in the exact same location, size and color of A's intention.

If a request for database connection is received by the control server, it calls the JDBC driver and connects to the Oracle database. Then, depending upon the content of the request, the control server will generate a different SQL statement, sending it to the database. The Oracle database gets the statement, processes it, and then returns the query result. The control server receives the result, and finally broadcasts it back to clients.

4. Runtime Environment Issues

To further understand the structure of VirtualBlock system, there are a few technical issues worth looking at in the runtime environment.

4.1 EAI

External Authoring Interface is a technique presented by the SGI. It allows developers to easily extend the functionality of the Cosmo Player VRML 2.0 browser and thereby build applications incorporating 3D, dynamic content. In essence, the EAI provides a method for developing custom applications that interact with, and dynamically update, a 3D scene. These outside applications "talk" to the VRML scene. For example, the user could enter a stock ticker into a Java applet which retrieves a real-time quote which would then be sent to the Cosmo Player. This real-time information can be sent through the EAI into the 3D scene, and generate visually compelling and intuitive market information. The EAI also allows the user to trigger events such as animations from outside the 3D world. The EAI is in the process of adoption into the VRML 2.0 specification.
Currently, it is considered an "Informative Annex", meaning browsers may include the EAI into their functionality and remain specification compliant.

In the VirtualBlock system, EAI is the mediator between the Java applet and the shared VRML world designers are working on. When the Java client receives a message from the Control Server broadcasting, the message comes with specifications of the size, position, and color of the block to be created. Through the EAI provided by the local VRML plug-in viewer, the Java applet would then be able to send the corresponding VRML sentence to construct the model in the VRML window. A clear understanding of how the EAI dynamically builds VRML files based on data received via Java applets and how, in turn, the application's data can be dynamically updated through the VRML interface, is of vital importance to the whole process.

![EAI diagram](image)

**Figure V-8 EAI diagram**

4.2 Network Protocol: TCP/IP vs. UDP
To exchange data between computers, the set of rules that both ends agree upon in order to understand each other is called a protocol. TCP/IP, standing for Transmission Control Protocol and Internet Protocol, is the common thread that ties the enormous Internet together. Most network software speaks TCP/IP, which could enable them to send information back and forth without a "language barrier" getting in the way. Java, as a highly network-support language, provides ability network connections through TCP/IP.

Data is broken into small packets, also called datagrams, and then wrapped up by network protocols before it is sent across the network. Since there are multiple routes between the beginning and the end point, the packets that make up a particular data stream may not all take the same route. Thus, they may not arrive at the same time or even in the same order. Some of them could even get lost along the way. TCP/IP has the feature to acknowledge receipt of data packets and request retransmission of lost packets. Moreover, TCP/IP allows the packets to be put back together on the receiving end in the same order they were sent on the sending end. We call TCP/IP a reliable protocol.

However, TCP/IP carries a fair amount of overhead. Therefore, if the order is not particularly important, and if the loss of individual packets will not completely corrupt the data stream, packets are sometimes sent without the guarantees that TCP/IP provides. This is accomplished through the use of the UDP protocol. The application of UDP is focused on Audio/Video streaming where the fluency of the transmission outweighs the completeness of the stream. Take audio for example, if one or two packets gets lost in
the transmission process, it will sound like the voice is a little jangled, but you can still catch the meaning. However, if the system tries to request the re-transmission of the lost packets, which may merely take several seconds, you will immediately find that the conversation is frustrating.

In the VirtualBlock system, I adopted the TCP/IP protocol for the exchange messages between client and server. The reason is obvious: collaborative designers simply cannot afford the loss of the accuracy of a model while it’s OK to wait for a few seconds. However, in the next stage of this project, it would be necessary to introduce audio and video conferencing on top of the shared VRML world. Then the UDP will have to be exploited in order to keep up the fluency of the real-time communication. Also, the bandwidth requirement will be much higher since literally “a picture is worth thousands of words” (in fact it is usually millions in terms of the bits that need transmitting).

4.3 Java Server and Client Socket

The communication between Java applet downloaded to a client's computer and the Java server program running on the control server layer is done with the help of Java socket technique. As we mentioned earlier, data is transmitted across the Internet in packets of finite size called datagrams. Each datagram contains a header and a payload. The header contains the address and port the packet is going to, the address and port the packet came from, and various other housekeeping information used to ensure reliable transmission. The payload contains the data itself. However, since datagrams have a finite length, it is often necessary to split the data across multiple packets and reassemble it at the destination. It is also possible that one or more packets may be lost or corrupted while in transit, and will need to be retransmitted, or that packets will arrive out of order. Keeping track of all this – splitting the data into packets, generating headers, parsing the headers of incoming packets, keeping track of what packets you have and have not received, readjusting the order of the incoming packet to recover its sequence, etc. – is a lot of work, and requires a lot of intricate software.

Fortunately, here come the sockets. Sockets allow the programmer to treat a network connection as another stream that bytes can be written onto or read from. A socket can perform seven basic operations:
- Connect to a remote machine
- Send data
- Receive data
- Close a connection
- Bind to a port
- Listen for incoming data
- Accept connections from remote machines on the bound port

Java's socket class, which is used here by both front application client applet and control server, has methods that correspond to the first four operations. The last three operations are only needed by a server that needs to wait for clients to connect to it.

In the VirtualBlock system, the control server layer has a Java server program running constantly on the server computer. It creates a server socket that listens to the client connection request. After the client downloads the Java applet to their local machine, a client socket will be created and then attempt to contact the server. When the server program receives the socket connection request of the client, an information channel is established between the server and the client. This is how data get moved around across the VirtualBlock system. The main type of data here being sent out and received are ASCII text messages. These describe the features of the VRML block to be created, the textual conversation from the chat box, the X and Y coordination of each stroke made in the shared drawing board, etc. If audio and video information is added into the system, it would also be transmitted in virtually the same way.

### 4.4 JDBC and SQL Database

JDBC, which stands for Java Database Connection technique, plays a significant role in the VirtualBlock system architecture. The JDBC API defines Java classes to represent database connections, SQL statements, result sets, database metadata, etc. It allows a programmer writing in the Java programming language to issue SQL statements and process the results. JDBC is the primary API for database access in the Java programming language.
The JDBC API is implemented via a driver manager that can support multiple drivers connecting to different databases. JDBC drivers can be either entirely written in the Java programming language so that they can be downloaded as part of an applet, or else they can be implemented using native methods to bridge to existing database access libraries. JDBC drivers fit into one of four categories:

1. The JDBC-ODBC bridge provides JDBC access via most ODBC drivers. Note that some ODBC binary code, and in many cases database client code, must be loaded on each client machine that uses this driver. So this kind of driver is most appropriate on a corporate network, or for application server code written in the Java programming language in a 3-tier architecture.

2. A native-API partly-Java technology-based driver converts JDBC calls into calls on the client API for Oracle, Sybase, Informix, DB2, or other DBMS. Note that, like the bridge driver, this style of driver requires that some binary code be loaded on each client machine.

3. A net-protocol all-Java technology-based driver translates JDBC calls into a DBMS-independent net protocol which is then translated to a DBMS protocol by a server. This net server middle-ware is able to connect its entirely Java technology-based clients to many different databases. The specific protocol used depends on the vendor. In general, this is the most flexible JDBC alternative. It is likely that all vendors of this solution will provide products suitable for Intranet use. In order for these products to also support Internet access, they must handle the additional requirements for security, access through firewalls, etc., that the Web imposes. Several vendors are adding JDBC drivers to their existing database middle-ware products.

4. A native-protocol all-Java technology-based driver converts JDBC calls into the network protocol used by DBMS directly. This allows a direct call from the client machine to the DBMS server, and is a practical solution for Intranet access. Since many of these protocols are proprietary, the database vendors themselves will be the primary source for this style of driver. Several database vendors have these in progress.

In the VirtualBlock system, the database I adopted is Oracle 7 for Windows NT. The JDBC driver provided by Oracle, Oracle's JDBC Thin Driver, is a type four driver that uses Java sockets to connect directly to Oracle. It provides its own implementation
of a TCP/IP version of Oracle's SQL*Net. Because it is written entirely in Java, this driver is platform-independent, which also ensures the platform independence of the applet. When the control server receives the request of database connection from the client, it calls the given JDBC driver, makes the connection to the Oracle database, sends out SQL queries, and retrieves the result.

The internal structure of the VirtualBlock database structure in Oracle contains five related entities: the VRML scene, the blocks, the movement of the blocks, the conversation texts, and the drawings. Every VRML scene can be viewed as a model project which consists of certain number of blocks. Each movement of each block, and the discussion and drawing associated with the move, is also recorded independently. Thus, to traverse the tree from the top node of a VRML scene, which could be seen as the design product, down to the bottom node of activities happened during the decision making of each piece of the model, one would have a clear understanding of the design. And that is the starting point of collaborative design.

Figure V-10 VirtualBlock database Entity-Relationship diagram
Chapter VI Conclusions

I believe that this thesis is a first step toward a new direction for collaborative design in the context of computer networks. The use of digital tools to facilitate design has great potential, and needs to be explored further.

There are two aspects in the computer-mediated design process that need to be studied in order to make it a more effective means of sharing design ideas and project development. The first aspect is on the technical side: the development of technology should follow the needs of the design professions to further fit itself into the design context. The second aspect involves a fundamental change in the way the design information is shared and communicated.

On the technology side, it seems we are already equipped with enough knowledge and practice on a basic level, including widespread CAD programs, growing Internet use, WWW and cross-platform programming languages, Virtual Reality techniques, etc. In fact, many projects have been done to explore the potential of existing tools in the computer mediated collaborative design process. In the next stage, there should be efforts to put the existing technology together and come up with an integrated system to support collaborative design. VirtualBlock is one of these explorations, and I see it as the prototype of the future application. Obviously, it needs improvements in almost every aspect of the system. The geometric modeling tools it provides are insufficient for complex design requirements. The limitation of a two-dimensional screen to represent three-dimensional objects must be overcome by a completely new interface design between humans and computers. Besides chat boxes and shared drawing boards, audio streaming and video conferencing would add more clarity in the communication process given ample network bandwidth and enough powerful computing ability.

The second aspect has to do with the use of technology in trying to formalize the way design information is represented and communicated. The traditional medium for designers to express their ideas has been to adapt to the new context in which digital media comes to play a central role in the whole game. Many have expressed their frustration when they try to work with computer technology, like CAD software, in design. The annoying fact that “we are being forced to think in the computer’s way” shows the
failure of the software engineers, who do not understand the way designers work, or
even the conflict between the characteristics such as rigidity and exactness featured by
computer technology, and the mysterious sub-consciousness and sometimes even irra-
tionality carried by design profession into the decision making process. While some de-
sign professionals are painstakingly adapting technology to the design context, shouldn’t
we designers also ask ourselves about changing our design formality and convention in
response to the rapidly changing world of technology? My personal opinion is that we
are facing the challenge of new tasks, new environments, and new tools. It is not simply
at matter of “having to think like a computer”. It is a matter of remodeling our way of
thinking to better harness these new tools, to better understand the new environment,
and as a result, to better deal with the new design tasks.
Bibliography


E. Branda, *Drawing Interfaces*, Massachusetts Institute of Technology, 1998


Nicholas Negroponte, Being Digital, Alfred A. Knofp, Inc., 1992


P. Rowe, Design Thinking, Cambridge, MA, MIT Press, 1987


Sonnenwald, D, Communication roles that support collaboration during the design process, Design Studies, 1994.


All illustrations by author except those credited in the illustration title.