VIRTUAL REALITY-BASED COLLABORATIVE ENVIRONMENT (VRCE) FOR EFFECTIVE PRODUCT DESIGN

BY

KAN HO YIN, B.Eng.

A Thesis Presented to
The Hong Kong University of Science and Technology
in Partial Fulfillment
of the Requirements for
the Degree of Master of Philosophy
in Industrial Engineering and Engineering Management

Hong Kong, August 1999

Copyright © by Kan Ho Yin 1999
AUTHORIZATION

I hereby declare that I am the sole author of this thesis.

I authorize the Hong Kong University of Science and Technology to lend this thesis to other institutions or individuals for academic purposes.

I further authorize the Hong Kong University of Science and Technology to reproduce the thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of research and education.
VIRTUAL REALITY-BASED COLLABORATIVE ENVIRONMENT (VRCE) FOR EFFECTIVE PRODUCT DESIGN

BY

KAN HO YIN, B.Eng.

APPROVED:

Dr. CHUAN-JUN SU  
SUPERVISOR
Department of Industrial Engineering and Engineering Management

Dr. VINCENT DUFFY  
CO-SUPERVISOR
Department of Industrial Engineering and Engineering Management

Prof. MITCHELL TSENG  
HEAD
Department of Industrial Engineering and Engineering Management

Dr. BENJAMIN YEN  
CHAIRMAN
Department of Industrial Engineering and Engineering Management

Dr. FUGEE TSUNG  
MEMBER
Department of Industrial Engineering and Engineering Management

Department of Industrial Engineering and Engineering Management  
August 1999
ACKNOWLEDGEMENTS

I greatly appreciate the informative and continuous guidance of my supervisor, Dr. Chuan-Jun Su. I also wish to acknowledge Dr. Vincent Duffy, my co-supervisor, for his constant support. Without their supervision and encouragement I would not be able to accomplish this research.

I extend my thanks to Dr. Benjamin Yen and Dr. Fugee Tsung for their invaluable support and suggestions. Special thanks go to Mr. Denil Chan for his technical support.
# Table of Contents

Title Page .................................................................................................................. i
Authorization Page .................................................................................................... ii
Signature Page .......................................................................................................... iii
Acknowledgments ..................................................................................................... iv
Table of Contents ..................................................................................................... v
List of Figures .......................................................................................................... viii
List of Tables ........................................................................................................... ix
Abstract .................................................................................................................... x

1. Introduction ........................................................................................................ 1
   1.1 Virtual Reality and its Applications on Product Design................................... 1
   1.2 Motivation ....................................................................................................... 3
   1.3 Research Objective ....................................................................................... 5

2. Literature Review ............................................................................................... 6
   2.1 Standards ....................................................................................................... 6
      2.1.1 DIS ............................................................................................................ 7
      2.1.2 VRML ...................................................................................................... 7
   2.2 Virtual Collaboration Systems (VCSs) .......................................................... 8
      2.2.1 dVISE ..................................................................................................... 8
      2.2.2 PIVOTAL ............................................................................................... 9
      2.2.3 Deneb .................................................................................................... 9
   2.3 Limitations of Current Systems .................................................................... 10
      2.3.1 Functionality and Customizability Issue .............................................. 10
      2.3.2 Portability Issue ................................................................................... 11
      2.3.3 Hardware Issue .................................................................................... 11
3. **Design of VRCE** ................................................................. 12

3.1 *Multi-view Collaborative Design Support* ................................................. 13
3.2 *Multi-layer Information Exchange* ............................................................ 14
3.3 *Multiple Opinions via Collaboration* .......................................................... 15
3.4 *Experimenting with Multiple Designs* ....................................................... 16
3.5 *Maturing Design Ideas over Time* ............................................................. 17
3.6 *Secure Collaboration Support* .................................................................. 17

4. **System Architecture** .............................................................................. 19

4.1 *Backbone standards* .............................................................................. 19
  4.1.1 Java ................................................................................................. 19
  4.1.2 VRML and EAI ................................................................................ 20
4.2 *Client-Server Architecture* ..................................................................... 21
4.3 *VRCE Engines* ...................................................................................... 23
  4.3.1 *Graphical Synchronization Engine* ................................................... 25
  4.3.2 *Message Transferring Engine* ............................................................ 25
  4.3.3 *Log Engine* ....................................................................................... 26
  4.3.4 *Security Servicing Engine* ................................................................ 27

5. **System Implementation** ........................................................................ 28

5.1 *Hardware Configuration* ........................................................................ 28
5.2 *Graphical Rendering and Networking tools* ............................................ 28
5.3 *Protocol* .................................................................................................. 29
5.4 *Server* ..................................................................................................... 31
  5.4.1 *Graphical Synchronization Engine* .................................................... 31
  5.4.2 *Message Transferring Engine* ............................................................. 33
  5.4.3 *Log Engine* ....................................................................................... 34
  5.4.4 *Security Servicing Engine* ................................................................ 34
5.5 *Client Program* ....................................................................................... 35
  5.5.1 *Graphical User Interface (GUI)* .......................................................... 35
  5.5.2 *Network Communication* .................................................................. 39
5.6 *Case Demonstration* .............................................................................. 39
5.7 *Comparisons between VRCE and existing Virtual Collaboration Systems (VCSs)* ...... 45
6. Discussion .................................................................................................................. 47
  6.1 A Pitfall .................................................................................................................. 47
  6.2 Virtual Collaboration System Customization Model ............................................ 49

7. Conclusion and Future Works .................................................................................. 47
  7.1 Conclusion ............................................................................................................ 52
  7.2 Future Works ........................................................................................................ 53

Reference ..................................................................................................................... 55

Appendices .................................................................................................................. 59
  A Browser and Plugin Compatibility for VRCE ........................................................... 59
  B Glossary .................................................................................................................. 60
List of Figures

Figure 4.1 Client-Server Architecture of VRCE ................................................. 22
Figure 4.2 VRCE Engines ............................................................................. 23
Figure 4.3 Support of VRCE Functional Design Concept by VRCE Engines ........ 24
Figure 5.1 Login Session of GUI of VRCE .................................................... 36
Figure 5.2 Collaborative Session of GUI of VRCE .......................................... 36
Figure 5.3 Functional Area of Collaborative Session ....................................... 37
Figure 5.4 A Shaded Joystick Model Loaded by Cosmetic Designer in VRCE ........ 40
Figure 5.5 Viewpoint Changing and Switching in VRCE ............................... 41
Figure 5.6 A Wireframe Joystick Model Loaded by Ergonomic Designer in VRCE ................................................................. 42
Figure 5.7 A Die for the Joystick Model Loaded by Manufacturing Engineer in VRCE ............................................................................. 43
Figure 5.8 Another New Joystick Design Loaded by Cosmetic Designer in VRCE... 43
Figure 5.9 VRCE Log File ............................................................................. 44
Figure 6.1 The “Inside-Out” View of Automobile Body Design in VRCE ............ 48
Figure 6.2 The “Outside-In” View of Automobile Body Design in VRCE ............ 48
Figure 6.3 Graphical Representation of Virtual Collaboration System

Customization Model ................................................................................... 49

Figure 6.4 Joystick and Automobile Body Design Examples Based on the

Conceptual Model ....................................................................................... 51
List of Tables

Table 5.1 VIP Ordinary-field........................................................................................................30
Table 5.2 VIP Pseudo-field........................................................................................................30
Table 5.3 Comparison between VRCE and three representative VCSs .........................45
Table A.1 Browser and Plugin Compatibility for VRCE .......................................................59
VIRTUAL REALITY-BASED COLLABORATIVE ENVIRONMENT
(VRCE) FOR EFFECTIVE PRODUCT DESIGN

BY
KAN HO YIN, B.Eng.

A Thesis Presented to The Hong Kong University of Science and Technology in Partial Fulfillment of the Requirements for the Degree of Master of Philosophy in Industrial Engineering and Engineering Management.

Hong Kong, August 1999

Abstract

Contemporary product design problems are inherently complex and involve many highly coupled sub-tasks that require multiple designers and engineers to work together collaboratively. Collaboration is needed for designers and engineers to get required information from others, identify and resolve design conflicts, and generate new ideas and design options.

Increasing complexity of engineering design problems have made collaboration an important part of engineering design rather than an auxiliary action. Although collaborative design is ubiquitous in industry, many in the engineering community recognize the fact that both design and collaboration are poorly understood activities in education, research and practice. Because of this poor understanding, collaborative design is still highly experience-based. Though the computer tools used for collaborative design exist, most of them only focus on supporting the industry that manufactures high-end, technology and technique oriented, expensive products. It is difficult to customize the functionality of these tools to suit small to medium-scale industry. Moreover, these tools are normally run on high-powered, dedicated machine (e.g., SGI ONYX) that small to medium-scale industry can hardly afford. Platform dependent of most of the existing tools also prevents collaborative design from becoming popular. As a result, there is a strong demand for a better understanding of the needs of collaborative design, and for an effective technology to support collaborative design practice. The technology should be accessible and beneficial to most of industries rather than a few specialized industries.

In response to the above need, this research attempts to generalize and propose key elements of collaborative design which are required by ordinary industries.

A Virtual Reality-based Collaborative Environment (VRCE), which is a Java and VRML based virtual reality environment with collaborative visualization, is developed to demonstrate the feasibility of the proposed research. VRCE emphasizes the use of multiple perspectives. These perspectives include multiple visual viewpoints, multiple information layers, multiple opinions, multiple designs as well as collaborating over time and security support. These allow virtual reality to be applied in the earlier, more creative, phases of the design process, rather than just as a walkthrough of the final design.
Chapter 1

Introduction

1.1 Virtual Reality and its Applications on Product Design

The term Virtual Reality (VR) is used by different people with different meanings. There are some people to whom VR is a specific collection of technologies, that is a Head Mounted Display, Glove Input Device and audio. Some other people stretch the term to include conventional books, movies or pure fantasy and imagination. One of the best definitions of VR to date comes from the book "The Silicon Mirage":

"Virtual Reality is a way for humans to visualize, manipulate and interact with computers and extremely complex data" [1].

The visualization part refers to the computer generating visual, auditory or other sensory outputs to the user of a world within the computer. This world may be a Computer Aided Design (CAD) model, a scientific simulation, or a view into a database. A user can interact with the world and directly manipulate objects within the world. Some worlds are animated by other processes, perhaps physical simulations, or simple animation scripts. Interaction with the virtual world, at least with near real time control of the viewpoint is a critical test for VR environments.

Some people object to the term Virtual Reality, saying it is an oxymoron. Other terms that have been used are Synthetic Environments, Cyberspace, Artificial Reality, Simulator Technology, etc. VR is the most common and sexiest. It has caught the
attention of the media. The applications being developed for VR run a wide spectrum, from games to architectural and business planning. Many applications are worlds that are very similar to our own, like CAD or architectural modeling. Some applications provide ways of viewing from an advantageous perspective not possible with the real world, like scientific simulators and telepresence systems, air traffic control systems. Other applications are much different from anything we have ever directly experienced before. These latter applications may be the hardest and most interesting systems. Visualizing the ebb and flow of the world's financial markets. Navigating a large corporate information base, etc [2].

Nowadays, industries have widely applied VR technologies to product design. Two of the most important applications of VR on product design are Virtual Prototyping (VP) and Simulation Based Design (SBD).

By utilizing VR and other advanced technologies, VP allows testing of alternative design ideas. The cost of testing by using VP is much smaller than that by using "real" prototypes [3]. At the same time VP provides high level of accuracy of design analysis. Generally VP of a product design involves three main steps [4]. The first step is the generation and modeling of a prototype. The second step is a validation of the prototype according to the initial specification. The final step is the feedback that includes request for minor changes, requirements for a more detailed model or other optimizations. By going through these steps, designers and engineers are allowed to investigate feasibility issue of product design at early stage according to the virtual prototype in a more effective manner. This is useful for preventing costly changes at later phases.
1 Introduction

The Simulation Based Design (SBD) incorporates VP in a more comprehensive manner by utilizing computer simulation techniques for product design using VP models. Simulation of the VP design is accomplished by the construction of a virtual prototype and virtual environment [5]. Users are able to immerse in virtual environment to interact with the virtual prototype with dynamic simulation. The interaction and immersion elements of VR with simulation support make SBD a more convenient and realistic representation of the simulated virtual prototype. Thus SBD has been proposed as an efficient and effective mechanism of evaluating large numbers of design alternatives for obtaining optimal design solutions.

1.2 Motivation

Three dimensional (3D) walk-throughs employing an “inside-out” perspective have traditionally been used for virtual environment [6]. This “inside-out” perspective allows a person to see the environment as if standing inside the environment and looking out at the surroundings. We believe this limits the usefulness of VR to only the final stages of the product design process, at which the 3D model of a product could be given to the VR environment to be displayed.

A long design development process has been a major problem of effective product design. This problem does not only occur in large-scale industries but also in small to medium-scale industries. The long design development process was attributed to iterating over many design problems resolving on meetings and phone calls while the problem resolving process is largely, if not completely, unsupported by computers.
1 Introduction

It is well known that collaboration is effective to reduce the number of iterations in design process. Collaborations are equally formal and informal. This means they spent approximately equal amounts of time at informal meetings with colleagues to chat about problems, as they did at formal meetings that were scheduled with colleagues, clients, and engineers. Furthermore, more work are accomplished in informal collaboration, where the emphasis was on the exploration of ideas, compared to formal collaboration, which mostly consisted of confirming designs that are brought to the meetings.

However traditional collaboration is geographically limited which colleagues are not able to collaborate and exchange their ideas if they are situated in different locations. Virtual collaboration is intended to solve this problem and it could consist of members from a company or from geographically distant locations (perhaps several thousand miles away). Thus the members could use modern networking technologies to collaborate, exchange their experience and ideas "virtually", as the same as "real" collaboration.

By utilizing the VR capabilities and technologies, virtual collaboration could be enhanced by incorporating the characteristics of VR in it. However, most of the existing VR-based, Virtual Collaboration Systems (VCSs) are not suitable for small to medium-scale industries due to their dedicated functionality which is difficult to customize, dedicated platform and high cost of investment in hardware. These limitations restrict the usage of VR-based collaboration to solve generic industries’ product design problems.
1 Introduction

1.3 Research Objective

As collaborative visualization has been considered valuable for reducing product design life cycle, our research attempts to elaborate collaborative design in industries, especially for small to medium-scale industries, by exploring the use of and interactive and collaborative VR environment, and portable and open distributed computing infrastructures. A Virtual Reality-based Collaborative Environment (VRCE) with a new set of comprehensive functionality design and customizability, low hardware requirement and portable platform has been developed to demonstrate the feasibility of the proposed research.

VRCE emphasizes the use of multiple perspectives. These perspectives include multiple visual viewpoints, multiple information layers, multiple opinions, multiple designs as well as collaborating over time and security support. These allow virtual reality to be applied in the earlier, more creative, phases of the design process, rather than just as a walkthrough of the final design. Moreover, by the power of VRCE, VR-based collaboration could then be widely promoted. Improved efficiency in product design and reduced iterations of product design life cycle through the generalized and proposed functional design concepts of VRCE are expected.
Chapter 2

Literature Review

As VR-based collaboration has made a great impact on reducing product design life cycle, the development of VCS has been of academic and commercial interests. VR, characterized by real-time simulation and presentation of, and interaction with a virtual world, is a powerful medium for people and computer communications. As its power, stand-alone application of VR would be too limited in extending its capabilities. Thus both of the academic and commercial sides have been seeking ways to make it more widely available by offering a widespread distribution of VR system and 3D interaction between people through Internet. As a result, people sit in Shanghai, Hong Kong, New York and Paris can collaborate through Distributed Virtual Reality (DVR), an emerging technology which brings people together to share the power of VR.

2.1 Standards

In order to achieve virtual collaboration across the global Internet by using DVR, transferring and updating of the contents of virtual environment through network are required. Certain standards are required to act as a common “language” between hosts to successfully transfer and update information. Among all the standards developed
so far, DIS (Distributed Interactive Simulation) and VRML (Virtual Reality Modeling Language) are two emerging standards for DVR.

2.1.1 DIS

Historically, DIS [7] has been focused on real time interactive military training applications. DIS is a network protocol based on IP multicasting for distributed interactive simulations. Its purpose is to facilitate military training at a cost far lower than training in a real battle field. It defines a number PDUs (Protocol Data Units), each of which is the format of the data in the packet describing a type of event. Different kinds of PDUs are transferred to all other network simulation participants in order to transfer the state of each object [8]. However, most PDUs are not suitable for ordinary DVR system and a permanent load is induced on the network even if object simulation states do not change [9]. As a result, DIS is not an ideal protocol for developing VRCE.

2.1.2 VRML

VRML is a file format for allowing computer users to visit and move through 3D virtual environments over Internet. VRML 1.0 [10], released in 1994, can only describe static environment whereas no dynamic behavior can be represented. In 1996, VRML 2.0 specification was publicized. Later on, the latest draft of the specification, VRML97 (April 97), replaced the old 2.0 version [11]. These new versions not only allow objects in a virtual world react to events but also support behavior specified in Java and JavaScript.
The capability of VRML in developing multi-user interactive environment is greatly enhanced with the help of External Authoring Interface (EAI) [12]. EAI, proposed by EAI Working Group, is an interface that allows an external environment to control the contents of a VRML browser window embedded in a web page. By utilizing Java’s networking capabilities and EAI, VRML could be used to construct global scale DVR systems and VCSs.

2.2 Virtual Collaboration Systems (VCSs)

Recognizing the needs of virtual collaboration, some groups (commercial groups mainly) have developed VCSs providing various features to achieve virtual collaboration. Representative systems include dVISE, PIVOTAL and Deneb.

2.2.1 dVISE

dVISE [13], developed by Division, is a product data visualization and simulation software that enables collaboration between designers and engineers team. It allows real time visualization and point of reference synchronization in live review session so that users are able to interact and redline three 3D models. Users could create and edit virtual mockups and combine different parts from several CAD systems into a mockup of the product for review. dVISE also supports not only creating real-time event-based behaviors to simulate product functions but also testing real time collision and clearance detection. Furthermore, it allows users to make and view virtual notes for design tracking, export mockups to MPEG movies and web-enabled
format for viewing. dVISE is available on Microsoft Windows NT, HP UNIX and IRIX platforms.

2.2.2 *PIVOTAL*

Centric Software’s PIVOTAL [14] is another important virtual collaboration solution which offers a collaborative prototyping and knowledge management tool. It focuses on product design so that designers and engineers can manipulate 3D models and own their personalized data views in order to get the information they need. User can perform model manipulation, real-time behavior test and design process simulation. Group discussion and collaboration are facilitated by real time audio, video and whiteboard conferencing with 3D annotation tools. PIVOTAL also supports model behavior activation, environment simulation and heterogeneous data incorporation. PIVOTAL runs on Microsoft Windows and UNIX platforms.

2.2.3 *Deneb*

Deneb Robotics Incorporation’s Virtual Collaborative Engineering (VCE) environment [15][16] is a VCS that enables real-time, interactive collaboration which facilitates discussion and analysis of simulations over global network among users. VCE allows users to interactively evaluate design concepts, manufacturing tooling, processes, and factory layouts in a dynamic simulation environment at geographically remote locations. VCE supports interactive model visualization, revision and manipulation during collaboration. Secure data transmission, simulation synchronization, interactive kinematics and collision detection, and independent
machine synchronization are also the core capabilities of VCE. VCE runs on Microsoft Windows NT and several UNIX platforms.

2.3 Limitations of Current Systems

The systems mentioned above are useful virtual collaboration tools that perform several significant functions in applying virtual collaboration, especially in product design. However, these systems tend to be limited in dedicated functionality, limited in customizability, lack of portability and demanding in hardware requirement, if not just one.

2.3.1 Functionality and Customizability Issue

Current VCSs have certain sets of functionality. Partly due to the lack of customer base, their functionality have been designed to focus on the industry that manufactures high-end, technology and technique oriented, and expensive products. The manufacturing of relatively low-end products was neglected even though they have a high value-added production. Furthermore, most of the existing systems are generally difficult to be customized functionally to suit the needs of common industries, especially small to medium-scale industries. This makes it hard for small to medium-scale industries to utilize virtual collaboration. With the rapid evolution and wide acceptance of VR technology, the opportunity of the development of a highly customizable VCS with a comprehensive functionality design is mature for the use of common industries. The development of such a system would greatly benefit
most of industries rather than just some high-end product oriented ones in terms of cost, efficiency, and effectiveness.

2.3.2 Portability Issue
The lack of portability of the current VCSs obstructs the popularity of collaboration. In a company or an organization, and/or throughout its branches, homogeneity of operating systems is not expected, or is even impossible. In order to facilitate and adapt virtual collaboration using the platform limited systems, certain operating systems need to be changed inside that company or organization. Changing of operating systems is a time consuming, risky and costly task that an industry would refuse to do. As a result, this seriously interferes with the use of virtual collaboration for product design life cycle reduction. Therefore a portable system is of utmost importance for effective virtual collaboration.

2.3.3 Hardware Issue
For nearly two decades, the introduction of first Silicon graphics workstation has promoted the growth of VR. While nowadays, it is noticed that in order to smooth the wide-spread growth of VR and virtual collaboration, systems should be developed so that they well functioned in affordable Personal Computers (PCs). Unfortunately, few existing VCSs support PC platform while most of them require high-end machines equipped with specialized hardware such as SGI ONYX. This high-end machine requirement sincerely prevents the virtual collaboration from being popularized despite the benefits that could be brought by using the environment.
Chapter 3

Design of VRCE

As most of the existing VCSs have dedicated functionality which is not able to cope with modern product design problems for common industry, existing functionality of virtual collaboration is refined and a new set of comprehensive functionality is generalized and proposed as the key capabilities of VRCE. This set of functionality design is focused on elaborating the traditional application of VR to collaborative product design for general industry that involves the use of multiple perspectives to support this process. From reviewing the literature and incorporating new ideas, these perspectives are generalized and include those that:

1. are produced as a result of applying differing viewing parameters to view a design.
2. are produced as a result of applying multiple layers of information for the tasks performed by individual users.
3. occur when multiple collaborators offer their opinions on the design.
4. occur when collaborators discuss alternative designs.
5. occur from design ideas maturing over time.
6. are produced as a result of collaboration in a secure environment.

In the following sections we will describe the functionality design of VRCE which is generalized from the above perspectives that include Multi-view Collaborative
Design Support, Multi-layer Information Exchange, Multiple Opinions via Collaboration, Experimenting with Multiple Designs, Maturing Design Ideas over Time and Secure Collaboration Support.

3.1 Multi-view Collaborative Design Support

One of the obvious advantages of VR is its ability to depict environments from an "immersive" perspective. That is, the viewers are placed in a position where they are physically immersed in the environment. This has been leveraged by many researchers to produce three-dimensional product spaces [17][18][19]. These implementations have been very successful because they offer designers the ability to view a design prototype before it is actually built.

Although the single viewpoint is useful in the evaluation of a pre-designed space, it is not necessarily the most appropriate viewpoint for the actual design process. For example, in a bicycle design process, an ergonomic designer would focus on the seat design while an engineer would focus on the design of gear and wheels. It would be difficult, from a wheel view to be able to position objects relative to the seat and still maintain a good sense of their relative position. Such a task is inherently easier if it were performed on the corresponding viewpoints. These corresponding viewpoints could be experienced and selected by collaborators freely. We call this notion as "Multi-view Collaborative Design Support".

Apart from multiple viewpoint experiment and selection, “Multi-view Collaborative Design Support” allows a collaborator to switch his or her viewpoint to one another’s viewpoints so that viewpoint sharing could be achieved. It is very useful to help
solving specific design problems. For instance, a manufacturing engineer points out a
design problem in the virtual environment. In order to make the problem
understandable, pure description would not be enough usually until all other
collaborators share the same viewpoint as manufacturing engineer has by viewpoint
switching in the virtual environment. This kind of viewpoint sharing and switching
function offers a valuable communication tool in virtual collaboration which has not
existed in any current VCS.

3.2 Multi-layer Information Exchange

Although multiple object perspectives have already been applied to VR in
applications such as architecture and virtual prototyping, little has been done to
provides a two-handed interface to allow a user to manipulate and activate a set of
filters to gain multiple perspectives over what is being visualized but does not extend
this to consider the dynamics of working amongst remote collaborators. It is our
conviction that multiple information representations are important in a collaborative
work environment. In this case, the collaborators may be working on the same
aggregate information but they are exploring decidedly different representations. For
example, in a computer game joystick design process, a cosmetic designer would
want to see the information as three-dimensional structures, such as a shaded model,
whereas a manufacturing engineer would want to see the same structure in terms of
manufacturability, such as feasibility of die making. Providing multiple information
layers over the same general model allows each participant to apply his or her
expertise to the problem at hand, by supplying them with the visual representations they are accustomed to interpreting.

3.3 Multiple Opinions via Collaboration

An important part of collaboration for product design is eliciting feedback from collaborators' colleagues. The roles which collaborators assume can dictate the actions they are capable of exercising in a collaborative virtual environment. For example, mortals (human beings) view the world from the "inside-out" and therefore could more easily perform fine manipulation of the environment such as finely moving, scaling, or rotating a sub-part of an object in a scene. On the other hand, deities (gods or goddess) view the world from the "outside-in" and therefore could more easily perform gross manipulation on objects in the environment.

Mortals and deities could assume the roles of apprentices and teachers respectively, or even clients and demonstrators. In such roles, the apprentice/client is in a position of learning/receiving a tour from the deity. Hence the deity can selectively turn certain mortal "powers" on or off as necessary. The deity may also have the ability to literally pick up mortals and bring them to the deity's perspective or even reduce their own perspective to that of the mortal's so that they may reside in the same space with the mortal. As their roles and understandings are different, opinions sharing between them is highly valuable during collaboration [21].
3.4 Experimenting with Multiple Designs

The application of VR has largely been dubbed a “rapid prototyping” tool. Although this is useful for finding design problems before a great deal of money is spent on actually building the structure, the use of VR still does not support the early creative design processes a lot; it only supports the end product. In order to have a gross idea of the product design, a CAD model must first be constructed. The CAD model only emerges after the cosmetic designer has, to a great extent, committed to the design. Changes made at this late stage will be costly in re-design time. Hence VR is only being applied after the tedium of drawing a CAD model is complete; it is not being used to support the creative task of design and problem solving. As VR seems to suit the problems solving in product design so well, it seems ironic that it should be used to support the creative phase of the process.

Our intent is to provide an environment that introduces VR-based collaboration early in the design process. This can be achieved through the following means. Engineers from various departments such as cosmetic design department, manufacturing and ergonomic department, will be allowed to express their concern regarding the design from their perspectives. This is useful for a cosmetic designer who is responsible for finalizing the overall shape of a product. If there is any potential problem found by other engineers, they can express their alternative design ideas to resolve the problem before finalizing the design. The high cost of modifying the design during the actual production phase can therefore be avoided.
3.5 Maturing Design Ideas over Time

The incorporation of the notion of time in the design environment is also necessary. That is, the virtual environment still persists after the participants leave. At a later stage, any participant may re-enter the space to perform more design work. This encourages informal collaboration to take place. It also affords the use of autonomous agents that can continue to perform tasks even when users have left the virtual environment. Since creativity does not follow a schedule, we believe that a collaborative environment that requires an a priori scheduling of its participants would be too limited. By provided with such a persistent virtual world, the designers may enter the world any time they have new inspirations for possible design solutions, without the system ever having to be shut down. The notion of persistent virtual spaces is greatly influenced by MUDs (Multi-User Dungeons), text-based multi-participant virtual environments that allows participants to enter, interact and leave at any time [22].

3.6 Secure Collaboration Support

System security and data transferring security are main security issues nowadays and high level of security is one of the success keys of a company. Yet most of the existing VCSs neglect the importance of these security issues during virtual collaboration where a large amount of valuable and confidential data flow such as flow of ideas, concepts, raw data and data files occurs. Regarding this issue, Security Collaboration Support is proposed in virtual collaboration. For instance, all unauthorized users should not be allowed to enter the collaboration environment in
order to achieve system security. For data transferring security, all the data and files transferred should be encrypted so that other unauthorized users could not access. If the collaboration is not taken place in a secure environment, the system would be easily hacked and the data would be lost. This causes a great economic loss, and more important, an intellectual property loss which would have a great negative impact on a company.
4.1 Backbone standards

The ultimate goal of VRCE is to provide a portable, low hardware demand and customizable system with a set of comprehensive functionality that facilitates virtual collaboration for product design. In order to achieve this goal, three open standards\textsuperscript{1} are considered to be the backbones of VRCE architecture. These backbone standards are Java, Virtual Reality Modeling Language (VRML), and VRML External Authoring Interface (EAI).

4.1.1 Java

From Sun Microsystems' description of Java [23]:

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

According to the above features, Java is considered to be very suitable for one of the backbone standards for constructing VRCE. Its powerful networking capabilities make it as a useful medium for building open and distributed network for VRCE. Its portability allows VRCE to become a system which could be run on most of popular
System Architecture

operating systems. Object Oriented (OO) approach of Java includes inheritance\(^2\) and polymorphism\(^3\) also enhances the extensibility and customizability of VRCE.

4.1.2 VRML and EAI

Another backbone standard being used in the VRCE is VRML. As described in Chapter Two - Literature Review, VRML is a popular and powerful 3D computer graphics standard. It is made by more than fifty computer graphics movers and shakers - including Microsoft, Apple, Silicon Graphics and Netscape [24]. Thus it has become the most widely accepted standard for displaying web-based virtual environment independent of a particular platform. Also the design of VRML has been focused on the use of low-end machine [11] that helps VRCE reach one of its main objectives: low requirement of hardware.

One more advantage of using VRML is its browser. VRML browsers interpret VRML files and make them visible so they can be navigated freely. Thus users are allowed to change his or her viewpoint continuously as they move their tracking and pointing devices. This capability helps us to achieve one of the VRCE functional design concepts, Multi-view Collaborative Design Support, as described in the Chapter Three - Design of VRCE.

Furthermore, VRML’s capability of developing multi-user environment is greatly enhanced by incorporating EAI. EAI provides a medium for the communication between VRML environment and an external environment such as Java. It consists of a set of functions which is responsible for accessing, adding and removing VRML nodes. It also allows the external source to send and get messages to and from VRML.
nodes. These capabilities offer data exchange and update between multiple users in
virtual environment. Thus the above factors make both VRML and EAI suitable for
being the backbones of VRCE.

4.2 Client-Server Architecture

In the proposed architectural design of VRCE, client-server architecture utilizing
broadcasting scheme is adopted. The reasons of using it are as follows:

1. By utilizing client-server architecture, clients with low computing power could
   connect to a more powerful server [25] so that computational requirement of
   client could be lowered. Thus the hardware requirement of client workstation
   could be reduced cost effectively.

2. Broadcasting client-server architecture is a less complex architecture than
   multicasting architecture that requires less complicated architectural design.
   Although the former is difficult to scale up for large number of simultaneous
   users (few hundred users) [26], it is considered to be appropriate for collaborative
   product design application which is not a labor intensive process.

3. As we use Java, VRML and EAI to construct the system, Java applet is used as
   network communication interface and graphical user interface. Java applet could
   not create a network connection to a machine other than the one from which the
   applet was downloaded [27], that is the server in this case. Thus this leads us to
   adopting the client-server architecture in the VRCE architectural design.

The client-server architecture of VRCE is shown in Figure 4.1.
With regard to the above design requirements, VNet [28], which is a multi-user VR system, is employed to be the prototype developing software of VRCE. It is a free software released under the GNU General Public License.

VNet adopts the client-server architecture and its server side is written by Java while its client side is written by Java, VRML and EAI. Source code accessibility of VNet, characterized by free software, offers flexible functional customization of VRCE for ordinary industries.

VNet uses the VRML Interchange Protocol (VIP) [29] as the protocol for the communication between client and server. VIP is a simple, compact and fast protocol for sending VRML fields over networks and runs over a reliable TCP/IP protocol. It is written in Java and is binary in nature. VRCE, as the same as VNet, utilizes VIP as networking protocol for the connections between client and server. These connections facilitate general administration control (Login and Quit) and information
distributions including text-based message exchanging, object manipulation, geometric changes of object and viewpoint changing.

When a client wants to distribute his or her information to other clients, a VIP stream connection is opened between client and server. The server receives information or updated information from the client, processed it and then distributes the information to other clients.

4.3 VRCE Engines

Inside the VRCE server, there are four engines that are responsible for different processes that facilitate virtual collaboration. These four engines, as shown in Figure 4.2, are Graphical Synchronization Engine, Message Transferring Engine, Log
Engine and Security Servicing Engine. They support different VRCE functional design concepts as shown in Figure 4.3. Their corresponding functionality are as

**Figure 4.3 Support of VRCE Functional Design Concept by VRCE Engines**
follows:

4.3.1 Graphical Synchronization Engine

For a distributed virtual world, an object’s status, including position and orientation, and an user’s viewpoint might change continuously. In order to make all of the client’s worlds synchronized with one another’s, there should be a graphical synchronization process to deal with all graphical update from clients. The Graphical Synchronization Engine is designed to accomplish this task. When certain client adds or removes objects in the VRCE, or he or she alters his or her viewpoint or object’s status, the updates are sent to this engine in the VRCE server for synchronization. After the Graphical Synchronization Engine receives the updates, it delivers those updates to certain or all clients at once. As a result, the whole VRCE can be kept synchronized. Graphical Synchronization Engine is designed to support three of the VRCE functional design concepts including *Multi-view Collaborative Design Support, Multi-layer Information Exchange* and *Experimenting with Multiple Designs* as shown in Figure 4.3.

4.3.2 Message Transferring Engine

The Message Transferring Engine takes care of listening and sending text-based messages to clients in order to achieve collaboration with multiple opinions. When any client sends a text-based message to other client(s), the message is first sent to this engine. Then it distributes the message to other client(s) and performs message filtering if necessary. Message filtering is a function inside this engine that filters the
recipients who would receive the message. Thus private discussion could be taken
place by using the Message filter.

The Message Transferring Engine is designed to facilitate one of the VRCE
functional design concepts, *Multiple Opinions via Collaboration* as shown in Figure
4.3.

4.3.3 *Log Engine*

The purpose of Log Engine is to break the time limitation of virtual collaboration so
that the virtual collaboration persists during the design project life cycle. The Log
Engine backs up all of the discussion contents by extracting the useful messages sent
through the VRCE server. Upon request from the later collaborators in the persisting
virtual collaborating environment, the discussion history is sent to the collaborator. It
serves as a valuable reference for later-joining collaborators so that they could keep
track of the collaboration during the design project life cycle. For instance, a designer
team in United States discusses a new toy design. After the office hours in the United
States, the designer team logs off the VRCE. Meanwhile, it is the beginning of Hong
Kong office hours, and the engineering team in Hong Kong logs in the persistence
VRCE. The Hong Kong team gets the previous discussion history of United States
team from the Log Engine. Thus the Hong Kong team knows the exact progress of
United States team and is able to continue the collaboration. Thus the time limitation
of virtual collaboration is broken.

The Log Engine is designed to support one of the VRCE functional design concepts,
*Maturing Design Ideas over Time* as shown in Figure 4.3.
4.3.4 Security Servicing Engine

The Security Servicing Engine is designed to solve the system security problem of virtual collaboration. It ensures only authorized collaborators from a certain department, division or organization could enter the VRCE. This enhances system security and facilities a "safe" collaborative environment for product design project. A new collaborator is allowed to register with the Security Servicing Engine to obtain a password for a certain project. He or she could then access VRCE of that project using this password. The Security Servicing Engine validates collaborator’s identification by password verification.

The Security Servicing Engine is designed to support one of the VRCE functional design concepts, Secure Collaboration Support as shown in Figure 4.3.
According to the proposed architecture of VRCE, a prototype implementation of VRCE is developed for feasibility study. It embodies our generalized and proposed functional design concepts for virtual collaboration. An example of virtual product design of computer game joystick is used for demonstration purpose.

5.1 Hardware Configuration

As one of the main focuses of our research is to overcome the limitations of virtual collaboration in dedicated and high-end hardware so that ordinary industries, even small to medium-scale companies are able to afford, we only utilized ordinary PCs for the VRCE development. For the central server, we have used a PC with Intel Pentium II 400 CPU as a server. For the client side, we have used several PCs with CPU ranged from Pentium 200MMX to Pentium II 400 with ordinary display cards. Those hardware configurations are very affordable nowadays and they give a very satisfactory performance for virtual collaboration in the VRCE.

5.2 Graphical Rendering and Networking tools

As mentioned before, Java, VRML and EAI are chosen as the developing backbones of VRCE. We have chosen VNet as the fundamental software for VRCE development
since it uses the open standards such as Java, VRML and EAI that meet our architectural design requirements. Also source code accessibility of it allows free and flexible customization of system by modifying the source code, even at the end user level. However VNet's original functionality is limited in general capabilities of virtual chat room. In order to achieve our proposed and generalized functional design concepts of VRCE, VNet's existing functionality was extended for VRCE prototype development.

VRML 2.0 is responsible for the client side graphical rendering of VRCE. Each client's world is delivered to a VRML browser via Hypertext Transfer Protocol (HTTP). Therefore collaborators who have a browser and a VRML plugin are able to access VRCE.

The networking of VRCE is programmed by Java that has powerful networking capabilities. Java's multi-threading capability enhances the performance of overall system since each of the clients is handled by his or her own thread and any individual thread does not need to be concerned about the rest of the threads execution at any point in the system [30].

The communication between VRML scene and Java relies on EAI. It operates by utilizing a set of classes which extends the Java platform for accessing VRML scene in VRCE.

5.3 Protocol

The communication between client and server over networks in VRCE is based on VRML Interchange Protocol (VIP) [29] running over a TCP/IP. All of the
communication through VIP is in form of a “Message”, which consists of three components: object, field, and value. The object component is the identification of each object that is assigned by VRCE server to each client. For the field component, there are two types: Ordinary-field and Pseudo-field. Ordinary-field is positively signed which represents the status the target object that should be modified. As Table 5.1 shows, according to the VIP specification, there are four fields that are used as the status representation of the object.

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POSITION</td>
<td>Object position that is broadcast by server to everyone except sender</td>
</tr>
<tr>
<td>ORIENTATION</td>
<td>Object orientation that is broadcast by server to everyone except sender</td>
</tr>
<tr>
<td>SCALE</td>
<td>Object scale that is broadcast by server to everyone except sender</td>
</tr>
<tr>
<td>NAME</td>
<td>Object name</td>
</tr>
</tbody>
</table>

**Table 5.1 VIP Ordinary-field**

Pseudo-field is negatively signed and is used to send Message containing action command but not involving the actual field value of the object. All of the Pseudo-fields are listed as Table 5.2.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QUIT</td>
<td>User disconnects from server</td>
</tr>
<tr>
<td>MESSAGE</td>
<td>User sends a text-based message to all other users through Server</td>
</tr>
<tr>
<td>ADD_OBJECT</td>
<td>User adds a new object to the scene</td>
</tr>
<tr>
<td>REMOVE_OBJECT</td>
<td>User removes an object from the scene</td>
</tr>
<tr>
<td>PRIVATE_MESSAGE</td>
<td>User sends a text-based message to certain user through server</td>
</tr>
<tr>
<td>CREATE_OBJECT</td>
<td>Server creates a new object identification</td>
</tr>
<tr>
<td>USER_INFO</td>
<td>Server sends the username of a new user to all other users</td>
</tr>
</tbody>
</table>

**Table 5.2 VIP Pseudo-field**
The value component is an encoded VRML field value that specifies the exact value that is sent.

5.4 Server

VRCE server was built based on VNet server with modifications. It is a multi-threaded server that communicates with client over networks based on VIP. VRCE server listens for incoming connections and setup communication socket. It creates a new thread for each successful connection it receives. The thread that created for each user deal with all of the communications between client and server during runtime. These communications are based on “Message” passing by VIP. Client’s thread receives all updates from its client and sends those updates to all other clients through server for data updating. Moreover, each thread is responsible for sending the updates from other clients from the server to its client. When the client quits, the socket connection would be broken.

In the VRCE server, four engines are implemented to support the VRCE architecture. They are Graphical Synchronization Engine, Message Transferring Engine, Log Engine and Security Servicing Engine.

5.4.1 Graphical Synchronization Engine

This engine is responsible for three graphical synchronization functions. The first function that it is responsible for is the synchronization of the avatar (3D object that represents user) adding, removing, position and orientation updates in the virtual environment. The engine is also responsible for the model (3D object that represents a
design idea) loading synchronization. The last function that it is responsible for is synchronizing users' viewpoint by request, which supports viewpoint switching.

When the engine receives updates from client, it firstly gives a synchronization priority of the update. The update that contains an action command sent by a negatively signed Pseudo-field gains first priority while the update that contains an object status sent by Ordinary-field gains second priority. The update with a first priority is sent to clients for synchronization first while the update with a second priority is sent to clients for synchronization afterwards.

The first function of the engine, avatar status synchronization, is a default function of VNet software. When a new user log in or log out VRCE, avatar that added or removed in the virtual environment could be synchronized in every user's screen except the sender. This is achieved by passing the ADD_OBJECT and REMOVE_OBJECT identifier of the Pseudo-fields of VIP to other users through the engine which gains first priority for synchronization in the engine. Position and orientation update of avatar could be synchronized in every user's screen except the sender. This is achieved by passing the POSITION and ORIENTATION identifier of the Ordinary-field of VIP to other users through the engine when a certain user changes his or her position or orientation. The Ordinary-field gains second priority for synchronization in the engine. Thus the status of avatar synchronization could be achieved in the virtual environment.

The second function of this engine is to synchronize the model added by user to everyone's screen, with no exception. All the models loaded in VRCE are in VRML1 or VRML2 format (both are .wrl). If a user loads a new model in the VRCE, he or she
could specify the location of the model by the Uniform Resource Locator (URL). An ADD_OBJECT identifier of the Pseudo-fields of VIP is then passed from the client to the engine and it gains first priority for synchronization. This action not only also indicates a model loading request but also specifies the model location from which it is loaded. The engine then sends the corresponding update to all users with the corresponding URL specified. A VRML node that represents the model would be created according the URL by the method createVrmlFromString in the client side and the model would be added to everyone’s screen.

The third function of the Graphical Synchronization Engine allows users to share their viewpoint. This means one user could request and switch to another users’ viewpoint so that they could have the same viewpoint perspective. When a certain user requests to switch to another user’s viewpoint, the requester could obtain the target user’s viewpoint data by requesting the POSITION and ORIENTATION identifier of the Ordinary-field of VIP from the engine. The engine would send viewpoint data with a second priority for synchronization. The requester’s viewpoint could then be synchronized as the same as the target user’s viewpoint by modifying the user viewpoint setting in the client side by the EAI fields EventInSFVec3f and EventInSFRotation which contains the position and orientation information of viewpoint respectively.

5.4.2 Message Transferring Engine

Functions of Message Transferring Engine are default functions of VNet software. It uses MESSAGE and PRIVATE_MESSAGE identifiers of Pseudo-field of VIP to
transfer all public and private text-based messages from sender to receivers. All the text-based messages that are received would be displayed in the chat output display in the receiver’s Java applet. The message would also be displayed in the sender’s Java applet.

5.4.3 Log Engine

The implemented Log Engine is responsible for backup all of the text-based messages that are sent from clients to the VRCE server. It extracts useful and meaningful information such as sender name, message contents, data and time from all of the “Message” that are sent through the VRCE server. It then writes the corresponding information to strings so that they could be further written into the log file. The extracted information in string type is then written in a log file byte by byte and the log file is placed in the VRCE server. Upon request from the later collaborators, the log file could be obtained from the VRCE server and a new browser window would be opened for displaying that log file via HTTP protocol.

5.4.4 Security Servicing Engine

The implemented Security Servicing Engine enables system security for VRCE by simple password verification. Only authorized user with correct password is permitted to enter the VRCE.
5 System Implementation

5.5 Client Program

The client program of VRCE is developed based on VNet client program with modification. It is composed of two parts, one is Graphical User Interface (GUI) and another is Network Communication. The former deals with all of the interactions between user and client program such as user inputs and virtual environment display. Whereas the latter handles all of the communication between client program and the VRCE server such as object status synchronization.

5.5.1 Graphical User Interface (GUI)

The GUI consists of a Java Applet and a VRML plugin window in which the Java applet is located at the bottom part of the browser and the VRML browser window is located at the top part of the browser. The Java applet deals with text-based (by inputting text) and command-based (by pressing button) interactions. Whereas the VRML plugin window displays the virtual environment and handles 3D graphical-based interaction.

There are two sessions of GUI. One is the Login session and another is the Collaborative Session.

The Login Session, as shown in Figure 5.1, performs security check and requests the username and password of the VRCE user. If the username and password entered are valid, the user is allowed to enter the second session, the Collaborative Session, as shown in Figure 5.2.
Figure 5.1 Login Session of GUI of VRCE

Figure 5.2 Collaborative Session of GUI of VRCE
The Collaborative Session is mainly composed of five functional area includes the Real Time 3D Model Incorporation, Collaborative Online Display, Log Viewer, Viewpoint Switching and Chatting.

![Figure 5.3 Functional Area of Collaborative Session](image)

**Real Time 3D Model Incorporation**

This area, as shown in Part 1 of Figure 5.3, contains a button and an input area which allows user to load model in the VRCE by specifying URL in the input area. This command is sent to and processed in Graphical Synchronization Engine of VRCE server. The engine then sends the update to other collaborators so that they could see the model and give real time comment that enriches the power of virtual collaboration.

**Collaborator Online Display**

This area, as shown in Part 2 of Figure 5.3, contains a list that displays the name of all collaborators who are currently logged into VRCE. This is an original capability of VNet.
Log Viewer

Log file of previous discussion and collaboration could be viewed by making use of the Log Viewer. If a user presses the "Log" button in this area, as shown in Part 3 of Figure 5.3, the discussion history of previous collaborators extracted by the Log Engine would be sent from the VRCE server to the client side. A new browser window would be opened for displaying the discussion log in the client side via HTTP protocol.

Viewpoint Switching

This area, as shown in Part 4 of Figure 5.3, contains a button that supports the user to request viewpoint switching. The requester could select a user on the Collaborator Online Display whose viewpoint the requester is interested in. Then the requester could click the "Switch View." button in the Viewpoint Filter area to request the corresponding data in the Graphical Synchronization Engine. The requester viewpoint would then obtain a viewpoint that is the same as the selected user’s viewpoint. Modification of the requester’s viewpoint setting is made on the client side by using the corresponding viewpoint data.

Chatting

Chatting is an original capability of VNet. The Chatting area consists of a chat output display, a chat input region and a “Private Msg.” button, as shown in Part 5 of Figure 5.3. The user is able to broadcast text-based message to all other users. The text-based message is sent to the Message Transferring Engine through VIP. The engine then
processes and delivers the message to other users. The user is also able to send private message to individual users by making use of the private message function so that all collaborators except the receiver can not see the corresponding message. The mechanism is similar to the broadcast message whereas the only difference is that the Message Transferring Engine only delivers the message to the receiver.

5.5.2 Network Communication

A Dispatcher of each client is responsible for the network communication with the VRCE server. The Dispatcher of each client receives updates from its client’s VRML browser window or Java applet. It then forwards the updates by its client thread through VIP to the VRCE server for processing and distributing the updates to other clients. On the other hand, the Dispatcher also receives updates from other clients through the VRCE server by its client thread and then forwards the updates to the GUI of its client for displaying.

5.6 Case Demonstration

The VRCE prototype allows a team of people to collaborate on a product design project in a virtual environment provided that collaborator has a computer connected to network with compatible browser and plugin as shown in Appendix A [31]. The network bandwidth requirement for running VRCE is low and the VRCE could be run from modem dial-up networking to various dedicated connections such as Integrated Services Digital Network (ISDN), Diver's Discount Network (DDN), etc with satisfactory performance.
A collaborator is able start the collaboration by firstly visiting the homepage of VRCE (http://iez059.ieem.ust.hk/) by a browser. The Login Session with a Java applet and a VRML browser window will appear. By entering valid username and password for security checking, the collaborator can then enter the Collaborative Session supported by the functional design concept *Secure Collaboration Support* and start the collaboration.

For instance, a cosmetic designer, an ergonomic designer and a manufacturing engineer in Hong Kong have logged in the VRCE. A cosmetic designer might load a model of his or her new joystick design in the VRCE for others to comment. Since the cosmetic designer concerns the appearance of joystick, a shaded model as shown in Figure 5.4 is loaded by him or her. All collaborators are able to view the model in multiple viewpoints and switch to others’ viewpoints with different perspectives by

*Figure 5.4 A Shaded Joystick Model Loaded by Cosmetic Designer in VRCE*
the functional design concept *Multi-view Collaborative Design Support* as shown in Figure 5.5. For example, from the top snapshot to the middle snapshot of Figure 5.5, the cosmetic designer changes his or her viewpoint to view the joystick model in different views. The manufacturing engineer then switches his or her viewpoint to the same as the cosmetic designer as shown in the bottom snapshot of Figure 5.5. They could also share opinions with each other by text-based message which supports the functional design concept *Multiple Opinions via Collaboration*. Their discussion is shown in the chat output display in Figure 5.5.

*Figure 5.5 Viewpoint Changing and Switching in VRCE*
The ergonomic designer might load a second layer of information which is a wireframe model of the joystick design in the VRCE by the functional design concept Multi-layer Information Exchange. The wireframe model allows the ergonomic designer to view the structure of the design clearly. He or she then points out some suggestions for the design, for example a much more rounded edge of the base because of safety reason, according to the second layer of information as shown in Figure 5.6.

![A Wireframe Joystick Model Loaded by Ergonomic Designer in VRCE](image)

**Figure 5.6** A Wireframe Joystick Model Loaded by Ergonomic Designer in VRCE

With the help of Multi-layer Information Exchange again, the manufacturing engineer might indicate the design of the cosmetic designer is feasible from the view manufacturing point of view by loading a third layer of information. This information is a die model that used to mold the corresponding joystick as shown in Figure 5.7.
After perceiving the others’ comments and ideas, the cosmetic designer might load another new joystick design in the VRCE as shown in Figure 5.8 which supported by the functional design concept *Experimenting with Multiple Designs*.

**Figure 5.7 A Die for the Joystick Model Loaded by Manufacturing Engineer in VRCE**

**Figure 5.8 Another New Joystick Design Loaded by Cosmetic Designer in VRCE**
Later on, other virtual team member such as engineering manager, sales manager and marketing manager in United States, with time difference, might join the virtual collaboration after the first group of collaborators left. They then view the log to understand the progress of product design project, which supported by the functional design concept *Maturing Design Ideas over Time* as shown in Figure 5.9. They might load alternative models and continue the collaboration for the joystick design project in this persistent virtual environment.

![VRCE Log File](http://iec553.ieem.ust.hk/previousLog.html)

**Figure 5.9 VRCE Log File**

With this kind of virtual collaboration, a virtual prototype of new designed joystick will emerge with the agreement of every expertise in a relative short period of time. The design could then be passed to production line for manufacturing with reduced number of iterations in the product design life cycle. Eventually, the time to market of the product is greatly reduced.
5.7 Comparisons between VRCE and existing Virtual Collaboration Systems (VCSs)

In order to summarize the strengths and weaknesses of VRCE, comparisons are made between VRCE and three representative VCSs, includes dVISE, PIVOTAL and Deneb. The comparisons are summarized as Table 5.3.

For portability, comparing with the above VCSs, VRCE is the most portable and it could be run on the most popular platforms nowadays.

<table>
<thead>
<tr>
<th></th>
<th>VRCE</th>
<th>DVISE</th>
<th>PIVOTAL</th>
<th>Deneb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform</td>
<td>Windows95, Windows98, Windows NT, Mac, UNIX, LINUX and IRIX</td>
<td>Microsoft Windows NT, HP UNIX and IRIX</td>
<td>Microsoft Windows and UNIX</td>
<td>Microsoft Windows NT and UNIX</td>
</tr>
<tr>
<td>Source Code Accessibility</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Multi-view Collaborative Design Support</td>
<td>Yes</td>
<td>Partially</td>
<td>Partially</td>
<td>Partially</td>
</tr>
<tr>
<td>Multi-layer Information Exchange</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Multiple Opinions via Collaboration</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Maturing Design Ideas over Time</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Experimenting with Multiple Designs</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Secure Collaboration Support</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 5.3 Comparison between VRCE and three representative VCSs
Source code accessibility enhances the functional customizability of VRCE for various applications while the above VCSs lacks this important capability.

VRCE are equipped with all six functional design concepts. On the other hand, the above VCSs have only partial Multi-view Collaborative Design Support since all of them lack the viewpoint switching capability while they allow users to have multiple viewpoints. For Maturing Design Ideas over Time, only dVISE is equipped with this functional design concept and it allows user to view and create notes for design tracking. The functional design concept, Secure Collaboration Support, or similar security concerns could not be found on all of the above VCSs.

All of the above VCSs are equipped with the functional design concept Multi-layer Information Exchange, Multiple Opinions via Collaboration and Experimenting with Multiple Designs.

All six functional design concepts that are critical for an effective VCS were implemented on the VRCE prototype. While according to the comparisons, the above VCSs are not equipped with all these six functional design concepts. Furthermore, portability and source code accessibility are two weaknesses of the above VCSs, compared with VRCE.
6.1 A Pitfall

With the six functional design concepts, the VRCE is valuable in enhancing efficiency and reducing iterations in product design life cycle. One of the VRCE functional design concepts, Multi-view Collaborative Design Support, is useful in accurate space positioning in product design processes. It is considered to be useful and easily implemented even in complex product design such as automobile body design. With the help of Multi-view Collaborative Design Support, automobile body designers and engineers are able to explore the design by different views. The "inside-out" view, as shown in Figure 6.1, is useful for inner body design of automobile whereas "outside-in" view, as shown in Figure 6.2, is useful for outer body design of automobile.

In order to completely utilize the power of VRCE, other VRCE functional design concepts are necessary to be considered in terms of applicability. For instance, as stated before, Multi-view Collaborative Design Support of VRCE is useful and could be embedded easily in automobile body design. Yet there is a demanding requirement of knowledge for embedding another functional design concept, Multi-layer Information Exchange, of VRCE in automobile body design. As a result, the time for system customization also increases. This is obvious when we consider embedding
Computational Fluid Dynamic\(^7\) (CFD) information layer into the existing VRCE for automobile body design. CFD contains complex data that a normal system developer

Figure 6.1 The "Inside-Out" View of Automobile Body Design in VRCE

Figure 6.2 The "Outside-In" View of Automobile Body Design in VRCE
would find difficulties to handle. Thus before considering to use VRCE, thorough understanding of both of the target product design processes and the six VRCE functional design concepts should be acquired. The following concept model further explains this pitfall.

6.2 Virtual Collaboration System Customization Model

![Diagram of virtual collaboration system customization model]

Figure 6.3 Graphical Representation of Virtual Collaboration System Customization Model

A conceptual model which is called Virtual Collaboration System (VCS) Customization Model is proposed to illustrate the relationship between product complexity, required knowledge for embedding function in VCS and the time for VCS customization. The graphical representation of this model is shown in Figure...
6.3. The input of this model is the product complexity (z). The outputs of this model are the required knowledge for embedding function in VCS (x) and the time for VCS customization (y). It is proposed that, in general, there is a positive correlation between x, y and z. Thus when z is high, x and y are typically high. This means a high product complexity would come with a high knowledge requirement for embedding function in VCS and a longer time for VCS customization. Since the product design life cycle time of a product using VCS is composed of the VCS customization time and the VCS usage time. A decrease in the VCS customization time reduces the product life cycle time greatly, especially when the first application of VCS on certain product where the VCS customization time involves a large proportion of the product design life cycle time. As there are repeats in using VCS on certain product, the proportion of the product design life cycle time that is involved by the VCS customization time shrinks. This implies the conceptual model is valid in the first few VCS applications, while the validity decreases as there are repeats in using VCS on certain product.

When applying different virtual collaborative product design processes by referencing this conceptual model, the required knowledge for embedding function in VCS and the time for VCS customization could be predicted and referenced. For instance, a joystick design as illustrated at the last chapter involves a relative low product complexity. Thus it is expected that a relative low demand on required knowledge for embedding function in VCS would accompany a relative shorter time for VCS customization of a joystick design. This is shown in the small box in Figure 6.4.
Therefore, the VCS could be applied on the joystick design process with relatively short customization time and high cost effectiveness.

![Figure 6.3 Joystick and Automobile Body Design Examples Based on the Conceptual Model](image)

On the other hand, an automobile body design as illustrated in the last section involves a relative high product complexity. Thus it is expected that a relative large demand on required knowledge for embedding function in VCS would accompany a relative longer time for VCS customization of an automobile body design. This is shown in the large box in Figure 6.4. This long customization time acts negative impact on the benefit of VCS (reduction in product life cycle time). Therefore special concern should be paid on whether VCS would result a reduction of total product life cycle time if it is adopted.
Chapter 7

Conclusion and Future Work

7.1 Conclusion

This research focuses on the design and development of VRCE, which is targeted to shorten the entire product design life cycle for industries, especially for small to medium-scale industries. VRCE is composed of a set of comprehensive functionality with customizability, platform independent in nature and could be run on ordinary hardware.

The comprehensive functionality of VRCE includes the six functional design concepts which are critical for effective product design in collaborative virtual environment. Functional customizability of VRCE extends its usefulness to various applications. Industries could make use of VRCE by its platform independence without spending high cost on changing operation systems. Low hardware requirement of VRCE reduces the investment cost on hardware for virtual collaboration.

The feasibility of the proposed functional design concepts and system architecture were tested and justified on the VRCE prototype. A pitfall of using VRCE for certain circumstances was illustrated by a proposed conceptual model.
7 Conclusion and Future Works

7.2 Future Works

There is a great potential for applying collaborative VR on product design and other areas such as virtual battlefield and medical visualization. Although a generalized set of VRCE functional design concepts were proposed and implemented in this research, extensions and improvements in functionality for effective product design are suggested for future research. Furthermore, suggestion is made on the extension of architectural design of VRCE for other applications.

1. Existing VRCE does not allow user to manipulate or change any parameter of model (3D object that represents a design idea) loaded in the virtual environment. It is suggested to introduce this kind of functionality in future research for product design so that user in virtual environment is able to change the shape, color, material or texture of the model in real-time. Thus, the time spent on virtual collaboration would be reduced because of the reduced cycle time of model editing and reloading.

2. Text-based discussion is adopted in the existing VRCE for product design while it might show certain unnatural and inconvenience in expressing idea. It is suggested that voice-based discussion could be incorporated in future research. By speaking native language or even dialect during discussion, collaborators would enhance the level of meaning expression and acquisition during virtual collaboration with each other. Text-based discussion could be used as a secondary aid of discussion.

3. Password verification is adopted for the functional design concept, Secure Collaboration Support in the existing VRCE for system security. In order to enhance the security level of VRCE, data security is suggested to be implemented...
in the existing VRCE in future research. This could be achieved by encrypting all the data, includes 3D model files and text-based messages that transferred through Internet during collaboration. The encryption could be achieved using Secure Sockets Layer (SSL) or other data encryption technologies. Thus all unauthorized person could not have the chance to intercept and read the data routed through the Internet. Detailed information about SSL could be referred to the Netscape Security Center [32].

Moreover, since certain tradeoffs between data encryption and system performance are supposed, the impact of imposing data security on the VRCE system performance is worthwhile to be investigated in future research.

4. Existing VRCE adopts broadcasting as the distribution scheme in architectural design which is a simpler approach. It works well in small-scale collaboration such as collaborative product design. While as the number of simultaneous users rises, for example in the case of virtual battlefield, unwanted traffic induced by broadcasting increase which would cause network flooding [33]. Thus, multicasting distribution scheme is suggested in future research to scale up existing VRCE for other applications such as virtual battlefield. Macedonia et al gave a clear discussion on using multicasting on large-scale virtual environment [34].
References


- 55 -
References


References


[22] Andrew Cowan, “Introduction to MUDs”,


[28] Stephen F. White, VNet,

References

[29] Stephen F. White, VRML Interchange Protocol – Specification,


[31] Jeff Sonstein, VNet version 1.1b1 F.A.Q.,

[32] Secure Sockets Layer,

[33] Macedonia, Michael and Zyda, Michael "A Taxonomy for Networked Virtual
     48-56.

[34] Macedonia, Michael R., Zyda, Michael J., Pratt, David R., Brutzman, Donald
     P. and Barham, Paul T. "Exploiting Reality with Multicast Groups: A Network
     Architecture for Large Scale Virtual Environments," Proceedings of the 1995
     IEEE Virtual Reality Annual Symposium, 11-15 March 1995, RTP, North
     Carolina, pp. 2-10.
### A. Browser and Plugin Compatibility for VRCE

<table>
<thead>
<tr>
<th>Tested and reported to work</th>
<th>Tested and reported to have problems</th>
</tr>
</thead>
</table>
| **Windows98**  
  - Communicator 4.6, CosmoPlayer 2.0  
  - Internet Explorer 4.0, pre-installed VRML viewer | **Windows 98 beta**  
  - "RC0", built-in VRML viewer |
| **Windows95**  
  - Communicator 4.05, CosmoPlayer 2.1 (beta)  
  - Communicator 4.04, CosmoPlayer 2.1 (beta)  
  - Communicator 4.0, CosmoPlayer 2.0  
  - Communicator 4.0, CosmoPlayer 1.0.2  
  - Navigator 3.03, WorldView 2.0  
  - Navigator 3.01, WorldView 2.0  
  - Navigator 3.01, CosmoPlayer 2.0  
  - Navigator 3.01, CosmoPlayer 1.0.2  
  - Internet Explorer 4.0, CosmoPlayer 2.1  
  - Internet Explorer 4.0, WorldView 2.0  
  - Internet Explorer 4.0, CosmoPlayer 2.0 | **Windows95**  
  - Communicator 4.03, CosmoPlayer 2.0  
  - Internet Explorer 4.0, IE control vrm12c (aka WorldView) |
| **Windows NT 4.0**  
  - Communicator 4.6, CosmoPlayer 2.0 | **Irix 6.x**  
  - Communicator 4.x |
| **Mac**  
  - 8600/300, Communicator 4.05, CosmoPlayer 2.1 alpha  
  - 7500/100, Communicator 4.01, CosmoPlayer 2.1 alpha  
  - 7200/75, Communicator 4.04, CosmoPlayer 2.0 | **Irix 6.3/6.4 (O2/Octane)**  
  - Navigator 3.0.1s, CosmoPlayer 1.0.2 |
| **Linux**  
  - FreeWRL |  |

Table A.1 Browser and Plugin Compatibility for VRCE
Appendices

B. Glossary

1. **Open Standard.** An open standard refers to a standard which has interoperability between hardware and software. The interoperability is defined by the industry at large and not one or two vendors.


2. **Inheritance.** In object-oriented programming, inheritance refers to the ability to derive new classes from existing classes. A derived class ("subclass") inherits the instance variables and methods of the base class ("superclass"). and may add new instance variables and methods. New methods may be defined with the same names as those in the base class, in which case they override the original one.

   Free On-Line Dictionary Of Computing

3. **Polymorphism.** In object-oriented programming, polymorphism refers to a programming language's ability to process objects differently depending on their data type or class. More specifically, it is the ability to redefine methods for derived classes.


4. **Free Software.** From the definition from Free Software Foundation (FSF). free software refers to the users' freedom to run, copy, distribute, study. change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:
Appendices

- The freedom to run the program, for any purpose (freedom 0).
- The freedom to study how the program works, and adapt it to your needs (freedom 1).
- The freedom to redistribute copies so you can help your neighbor (freedom 2).
- The freedom to improve the program, and release your improvements to the public, so that the whole community benefits. (freedom 3).


5. **Satisfactory Performance.** From professional judgement, a satisfactory performance of virtual collaboration refers to the responses (include graphical and text-based) in virtual collaboration is free from discontinuity based on human visualization. Quantitative measurement for satisfaction such as the relationship between time lag of responses and hardware and/or network bandwidth requirement(s) is suggested for future work.

6. **Multi-threaded.** Sharing a single CPU between multiple tasks (or "threads") in a way designed to minimize the time required for switching threads. This is accomplished by sharing as much as possible of the program execution environment between the different threads so that very little state needs to be saved and restored when changing thread.

Free On-Line Dictionary Of Computing


7. **Computational Fluid Dynamic (CFD).** CFD is used for basic studies of fluid dynamics, for engineering design of complex flow configurations, and for
Appendices

predicting the interactions of chemistry with fluid flow for combustion and propulsion. It is also used to interpret and analyze experimental data and to extrapolate into regimes that are inaccessible or too costly to study.

The Naval Oceanographic Office Major Shared Resource Center

[www]: http://www.navo.hpc.mil/CFD.html