ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/222705051

An Internet virtual reality collaborative environment for effective product design

Article *in* Computers in Industry · June 2001

DOI: 10.1016/S0166-3615(01)00093-8

CITATIONS
125

READS
903

3 authors, including:

Vincent G. Duffy
Purdue University
198 PUBLICATIONS **2,538** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

IE556 Job Design (also PSY 556) View project

Gamification View project

All content following this page was uploaded by Vincent G. Duffy on 27 December 2018.

The user has requested enhancement of the downloaded file.

ELSEVIER

Computers in Industry 45 (2001) 197–213

COMPUTERS IN INDUSTRY

www.elsevier.nl/locate/compind

An Internet virtual reality collaborative environment for effective product design

H.Y. Kan, Vincent G. Duffy[*,1], Chuan-Jun Su

Department of Industrial Engineering and Engineering Management, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong SAR, PR China

Received 27 January 2000; accepted 24 January 2001

Abstract

This paper describes an Internet-based virtual reality collaborative environment called virtual reality-based collaborative environment (VRCE) that was developed using VNet, a free software, Java and Virtual Reality Modeling Language (VRML) to demonstrate the feasibility of collaborative design for small to medium size companies that focus on a narrow range of low cost products. Infrastructure, user functions and operational characteristics are described, and a case illustrates the use of this system that incorporates six key design elements needed for a successful virtual collaboration system (VCS). This new VRCE system is compared to existing commercial systems. Discussion highlights the potential pitfalls in using such a system without careful consideration of the knowledge required, product complexity and required system customization time. © 2001 Published by Elsevier Science B.V.

Keywords: Virtual reality-based collaborative environment (VRCE); Virtual Reality Modeling Language (VRML); Virtual reality (VR)

1. Introduction

A major problem in achieving effective and timely product designs and product innovations has been the long design and product development process. This problem does not only occur in large-scale industries but also in small to medium-scale industries. The long design development process can be attributed to the need for many design iterations, and problems typically are resolved in meetings or with many phone calls. Many times, in small to medium size companies, the problem solving process is largely, if not completely, unsupported by computers.

Many times work is accomplished in informal collaboration, where the emphasis is on the exploration of ideas, compared to formal collaboration, which mostly consists of confirming designs that are brought to the meetings. However, traditional collaboration is geographically limited. Colleagues are not easily able to collaborate and exchange their ideas if they are situated in different locations. Virtual collaboration is intended to solve this problem. The members can use modem networking technologies to collaborate and exchange their experience and ideas "virtually".

Virtual collaboration could be enhanced by incorporating virtual reality (VR) technology. However,

[*] Corresponding author. Tel.: +852-2358-7116; fax: +852-2358-0062.
E-mail addresses: vduffy@ust.hk, vduffy@ie.wisc.edu (V.G. Duffy).

[1] Parts of the manuscript were prepared while the author was serving as a Visiting Assistant Professor in the Department of Industrial Engineering, University of Wisconsin-Madison, Madison, WI, USA.

most of the existing VR-based, virtual collaboration systems (VCSs) are not suitable for small to medium-scale industries due to dedicated functionality that is difficult to customize. Dedicated functionality is the part of some systems that are targeted to serve a certain industry. This type of system is found especially in companies that manufacture high-end, technology and technique oriented, expensive products. Yet that kind of functionality cannot facilitate solving the design and manufacturing problems of industry in general. Many times there is a dedicated platform and requirement for high investment in hardware. These limitations restrict the use of VR-based collaboration and make it difficult when considering more general product design problems.

1.1. Virtual reality applications in 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. Using VP the cost of testing is much lower than when using "real" prototypes [1]. At the same time VP provides a high level of accuracy of design analysis. Generally, VP of a product design involves three main steps [2]. 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 requests for minor changes, requirements for a more detailed model or other optimizations. By going through these steps, designers and engineers are able to investigate the feasibility of a design in a more effective manner at an early stage according to the virtual prototype. This is useful for preventing costly changes at the latter phases.

The 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 [3]. Users can be immersed in the design in a virtual environment. Users interact with the virtual prototype. Some virtual environments contain dynamic simulations. The interaction and immersion elements of VR with simulation support make SBD a more convenient and realistic representation of the virtual prototype. SBD has been used as an efficient and effective mechanism to obtain optimal design solutions when evaluating a large number of design alternatives.

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 [4].

The visualization part refers to the computer generation of 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 virtual world and directly manipulate objects within the world. Some virtual worlds are animated with 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.

1.2. Research objective

This research attempts to facilitate collaborative design in industries, especially for small to medium-scale industries, by exploring the use of interactive and collaborative VR environments using inexpensive software, and portable and open distributed computing infrastructures. A virtual reality-based collaborative environment (VRCE) with comprehensive functionality and customisability, low hardware requirements and a portable platform has been developed to demonstrate the feasibility of the proposed research.

Three dimensional (3D) walk-throughs employing an "inside-out" perspective have traditionally been used for virtual environments [5]. 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.

VRCE emphasizes the use of multiple perspectives. These perspectives include multiple visual viewpoints, multiple information layers, multiple collaborators input, multiple designs, as well as collaboration 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 walk-through of the final design.

2. Literature review

In order to achieve virtual collaboration across the Internet using distributed virtual reality (DVR), transferring and updating of the contents of a virtual environment through a network is 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, distributed interactive simulation (DIS) and Virtual Reality Modeling Language (VRML) are two emerging standards for DVR.

2.1. *DIS and VRML*

Historically, DIS [6] has been focused on real time interactive military training applications. DIS is a network protocol based on IP multicasting for distributed interactive simulations. It defines a number of protocol data units (PDUs), 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 [7]. However, most PDUs are not suitable for an ordinary DVR system and a permanent load is induced on the network even if object simulation states do not change [8]. As a result, DIS is not an ideal protocol for developing VRCE.

VRML is a file format for allowing computer users to visit and move through 3D virtual environments over the Internet. VRML 1.0 [9], released in 1994, can only describe a static environment and no dynamic behavior can be represented. In 1996, the VRML 2.0 specification was publicized. VRML97 (April 97), replaced the old 2.0 version [10]. These new versions allow objects in a virtual world to react to events and support behavior specified in Java and JavaScript.

The capability of VRML in developing a multi-user interactive environment is greatly enhanced with the help of external authoring interface (EAI) [11]. EAI, 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 can 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.

2.2.1. *dVISE*

dVISE [12] is product data visualization and simulation software that enables collaboration between designers and the engineers. It allows real time visualization and point of reference synchronization in a live review session so users are able to interact and redline 3D models. Users can create and edit virtual mockups and combine different parts from several CAD systems into a mockup of the product for review. dVISE supports not only creation of 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*

PIVOTAL [13] 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. The user can perform model manipulation, real-time behavior tests and design process simulations. 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 Corporation's virtual collaborative engineering (VCE) environment [14,15] is a VCS that enables real-time, interactive collaboration which facilitates discussion and analysis of simulations over a global network. 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 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 important functions in applying virtual collaboration, especially in product design. However, these systems tend to be limited in customisability, lack portability and have demanding hardware requirements.

2.3.1. Functionality and customisability issue

Partly due to the lack of customer base, their functionality has been designed to focus on the industrial customers that manufacture high-end, high technology, and expensive products. Furthermore, most of the existing systems are generally difficult to customize. This makes it difficult for small-scale industrial companies such as manufacturers of speakers, lamps, computer accessories such as a mouse or fan, and video game accessories such as joystick and steering wheel controllers, to get the benefits of such systems. The global video game industry averaged 40% growth annually in worldwide sales in the early 1990s and continues to be an important part of the toy industry [16]. There has been a rapid evolution and wide acceptance of VR technology, and there is now a greater need for the development of a highly customizable VCS with comprehensive functionality.

2.3.2. Portability issue

In order to facilitate virtual collaboration using the platform limited systems, certain operating systems need to be changed inside that company or organization. Changing operating systems is a time consuming, risky and costly task. This seriously interferes with the potential 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, since the introduction of the first Silicon Graphics (SGI) workstation, many people have promoted and anticipated the expanded use of VR in industry. In small to medium size companies, however, most use a PC. Unfortunately, few existing VCSs support a PC platform. Most still require the use of specialized computer hardware such as the SGI ONYX II. This high-end computing requirement and initial cost has prevented virtual collaboration from being popularized, despite the potential benefits.

3. The design of VRCE and user functions

In this section, a set of comprehensive functions included in the design of VRCE is described. VRCE incorporates six design elements necessary for a successful VCS. The proposed VRCE supports the following user functions:

- Multiple viewing parameters.
- Selecting from multiple layers of information available for the tasks to be performed by individual users.
- Participation by multiple collaborators in the design.
- Discussion of alternative designs in real time.
- Archiving (logging) of discussion which allows maturing of design ideas over time.
- Collaboration in a secure environment.

When different design experts log into the VRCE and they load their respective design representations into the system, the system allows users not only to view the model in multiple viewpoints, but also to switch to each other viewpoints by the function — *Multi-view Collaborative Design Support*. During their viewpoint changing and switching, they can share information and their experience through text-based discussion. The system will deliver the comments and messages to the intended (or all) recipients. This function is called *Multiple Opinions via Collaboration*. The system allows multiple model loading so as to achieve *Multi-layer Information Exchange* and *Experimenting with Multiple Designs*. Each perspective in product design has different layers of information, such as appearance, manufacturing requirements and internal structure. The sharing and exchange of this information between different experts is extremely important for effective product design.

Users can load any model with different representations any time, so different information layers can be presented to avoid potential design conflicts. Users can also load any new design idea into the system anytime for design comparison and innovation. The system records all users' conversation history by generating logs. The logs contain valuable information of the past design process which can be referenced by the other users. This achieves *Maturing Design Ideas over Time*.

4. System architecture

The ultimate goal of VRCE is to provide a portable, low hardware demand and customizable system with a set of comprehensive functions that facilitates virtual collaboration for product design. In order to achieve this goal, three open standards are considered for the backbone of the VRCE architecture. These standards are Java, VRML, and VRML EAI.

For VRCE, a client-server architecture utilizing a broadcasting scheme is adopted. The reasons are as follows:

- By utilizing client-server architecture, clients with low computing power can connect to a more powerful server [17] so that the computational requirement of the client can be lowered. Thus, the hardware requirement of the client workstation can be reduced.
- A broadcasting client-server architecture is a less complex architecture than a multicasting architecture. Although the former is difficult to scale up for large numbers of simultaneous users (few hundred users) [18], it is considered to be appropriate for collaborative product design that would typically be done in small teams in small-medium sized firms.
- As we use Java, VRML and EAI to construct the system, a Java Applet is used as the network communication interface and graphical user interface. A Java Applet cannot create a network connection to a machine other than the one from which the Applet was downloaded [19]. That is the server in this case.

VNet [20] a multi-user VR system, is used as the prototype for developing software for the VRCE. It is a free software released under the GNU General Public License. VNet adopts the client-server architecture. Its server side is written in Java while its client side is written in Java, VRML and EAI. Source code accessibility of VNet, the free software, offers flexible functional customization of VRCE.

5. System implementation

A prototype of VRCE was developed to determine feasibility. It embodies the proposed functional design concepts for virtual collaboration. Virtual product design of a computer game joystick is used for demonstration purposes.

5.1. Hardware configuration

As one of the main focuses of the system is to overcome the limitations of virtual collaboration in dedicated and high-end hardware, ordinary PCs are used 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 ranging from Pentium 200MMX to Pentium II 400 with ordinary display cards. Those hardware configurations are very affordable and give satisfactory performance[2] for virtual collaboration in the VRCE.

5.2. Graphical rendering and networking tools

VNet is the primary software for VRCE development since it uses the open standards such as Java, VRML and EAI that meet our architectural design

[2] Satisfactory performance. For the purpose of this research, it refers to the ability of the graphical and text-based position of the virtual collaboration system to be free from discontinuity that might be observed by a user.

Table 1
VIP ordinary-field

Identifier	Description
POSITION	Object position that is broadcast by server to everyone except sender
ORIENTATION	Object orientation that is broadcast by server to everyone except sender
SCALE	Object scale that is broadcast by server to everyone except sender
NAME	Object name

requirements. Also source code accessibility of it allows free and flexible customization of the system by allowing modification of the source code, even at the end user level. However, the original functionality of VNet is limited in terms of the general capabilities of the virtual chat room. In order to achieve the proposed functional design concepts of VRCE, VNet's existing functionality was extended during the 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 such as Netscape 4.5 via Hypertext Transfer Protocol (HTTP). Therefore, collaborators who have a browser and a VRML plug-in such as Cosmo Player 2.0 are able to access VRCE.

The networking of VRCE is programmed in Java because of its powerful networking capabilities. Java's multi-threading[3] capability enhances the performance of the 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 [21]. The communication between the VRML scene and Java relies on EAI. It operates by utilizing a set of classes that extends the Java platform for accessing the VRML scene in VRCE.

5.3. Protocol

The communication between client and server over networks in VRCE is based on VRML Interchange Protocol (VIP) [22] running over a TCP/IP. All of the communication through VIP is in the 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 the 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 of the target object that should be modified. As Table 1 shows, according to the VIP specification, there are four fields that are used as the status representation of the object.

Pseudo-field is negatively signed and is used to send the "Message" containing an action command but does not involve the actual field value of the object. All of the pseudo-fields are listed in Table 2. The value component is an encoded VRML field value that specifies the exact value that is sent.

5.4. Server

The VRCE server was built based on the VNet server, with modifications. It is a multi-threaded server that communicates with the client over networks based on VIP. The VRCE server listens for incoming connections and sets up a communication socket. It creates a new thread for each successful connection it receives. The thread created for each user deals with all of the communications between client and server during runtime. These communications are based on the "Message" passed by VIP. Client's thread receives all updates from its client and sends those updates to all other clients through the 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.

These connections facilitate general administration control (Login and Quit) and information distributions including text-based message exchange, object

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

Table 2
VIP pseudo-field

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

manipulation, geometric changes of an object and viewpoint change. 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, processes it and then distributes the information to other clients.

5.5. VRCE engines

Inside the VRCE server, there are five engines that are responsible for different processes that facilitate virtual collaboration. These engines include the Graphical Synchronization Engine, Message Transferring Engine, Log Engine, Security Servicing Engine and Knowledge Engine. They support different VRCE functional design concepts as shown in Fig. 1. Their corresponding functionality is as follows:

5.5.1. Graphical Synchronization Engine

The 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 Fig. 1.

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. This 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 a client, it first 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 the VNet software. When a new user logs in or logs out of VRCE, the avatar that was added or removed from the virtual environment can 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 the avatar can 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 avatar synchronization is achieved in the virtual environment.

The second function of this engine is to synchronize the model added by the user to everyone's screen. 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

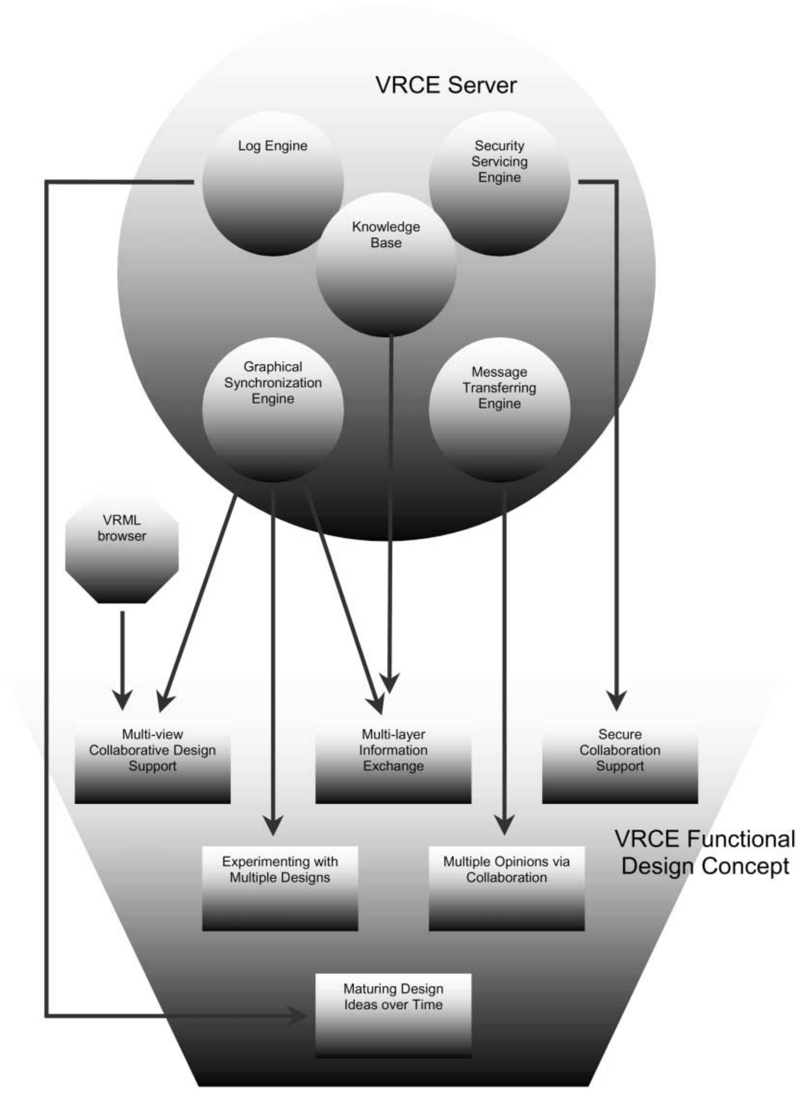

VRCE Server
Log Engine
Security Servicing Engine
Knowledge Base
Graphical Synchronization Engine
Message Transferring Engine
VRML browser
Multi-view Collaborative Design Support
Multi-layer Information Exchange
Secure Collaboration Support
VRCE Functional Design Concept
Experimenting with Multiple Designs
Multiple Opinions via Collaboration
Maturing Design Ideas over Time

Fig. 1. Support of VRCE functional design concept by VRCE engines.

first priority for synchronization. The engine then sends the corresponding update to all users with the corresponding URL specified. A VRML node that represents the model is created according to the URL by the method createVrmlFromString on the client side, and the model is added to everyone's screen.

The third function of the Graphical Synchronization Engine allows users to share their viewpoint. This means one user can request and switch to another users' viewpoint so that they can have the same viewing perspective. When a certain user requests a switch to another user's viewpoint, the requester can 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 with the target user's viewpoint by modifying the user viewpoint setting on the client side with the EAI fields EventlnSFVec3f and EventlnS-

FRotation that contain the position and orientation information of viewpoint, respectively.

5.5.2. Message Transferring Engine

The Message Transferring Engine is designed to facilitate one of the VRCE functional design concepts, *Multiple Opinions via Collaboration* as shown in Fig. 1. When any client sends a text-based message to other client(s), the message is first sent to this engine. Functions of the Message Transferring Engine are default functions of the VNet software. It uses MESSAGE and PRIVATE_MESSAGE identifiers of a pseudo-field of VIP to transfer all public and private text-based messages from senders 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.

5.5.3. Log Engine

The Log Engine is designed to support one of the VRCE functional design concepts, *Maturing Design Ideas over Time*, as shown in Fig. 1. It serves as a valuable reference for late-joining collaborators so that they can keep track of the collaboration during the entire life cycle of the design project.

The Log Engine, responsible for backup and archiving of all of the text-based messages that are sent from clients to the VRCE server, 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 can be further written into the log file. Upon request from the later collaborators, the log file can be obtained from the VRCE server and a new browser window can be opened for displaying that log file via HTTP protocol.

5.5.4. Security Servicing Engine

The Security Servicing Engine is designed to solve the system security problem for virtual collaboration. It ensures only authorized collaborators from certain departments, divisions or organizations can enter the VRCE. This enhances system security and facilities a "safe" collaborative environment for product design projects. A new collaborator is allowed to register with the Security Servicing Engine to obtain a password for a certain project. 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 Fig. 1.

5.5.5. Knowledge Engine

A description of previous work in Knowledge Engine development for industrial training by the IEEM/VR research group at HKUST in Hong Kong can be found in [23,24]. However, Knowledge Engine development is outside the scope of VRCE. A knowledge-server that appears highly informative and suitable for web-based knowledge management for distributed design is shown in [25]. The web-based architecture appears compatible with the VRCE web-based system. Please see [25] for further details of an example of a Knowledge Engine that could possibly be modified and incorporated into the VRCE system. Well-defined knowledge bases and Knowledge Engines should enhance the effectiveness and efficiency of VRCE.

The Knowledge Engine mainly supports *Multi-layer Information Exchange*. In order to resolve the design conflict, different layers of information are needed during the product design process. Since VRCE is designed to solve product design problems in different industries, consideration should be given to the fact that different sets of knowledge must be plugged into the VRCE Knowledge Engine so as to customize it, and enable solving the design problems of different industries. Knowledge requirements for different scenarios are discussed in more detail in Section 7.

5.6. Client program

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

The GUI consists of a Java Applet and a VRML plug-in 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. This Java Applet has text-based (text input) and command-based (button pressing) interactions. The VRML plug-in 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 performs a 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. The Collaborative Session is mainly composed of five functional areas and includes the real time 3D models, collaborative online display, view of the log, viewpoint switching and chatting.

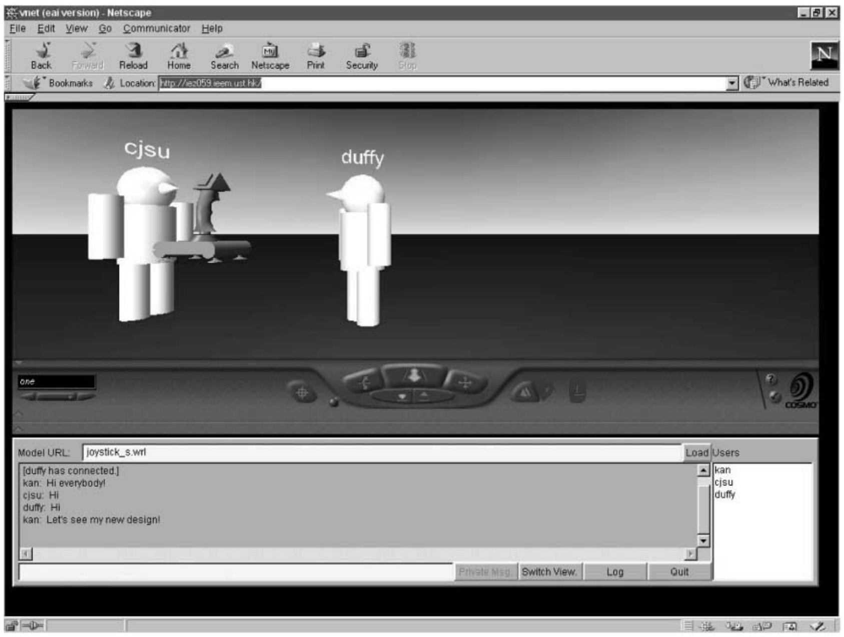
5.7. Operational characteristics and a case demonstration

A collaborator is able start the collaboration by visiting the homepage of VRCE with an Internet browser. Before a user can enter the VRCE, the system requires the user to enter a password for authentication during the Login Session. The Login Session with a Java Applet and a VRML browser window will appear. Then a user would enter the core session of VRCE. In this session, user can become immersed in a VRML-based virtual environment. The user can navigate in the virtual environment freely and interact with different designs in the upper frame. In the lower frame, there is a Java Applet which is an interface that allows the user to

- Load a model in the virtual environment by entering correct path of the model.
- Share their ideas and experience by typing messages in the text area.
- Switch viewpoints by selecting the targeted collaborator.
- View the log window with conversation history.

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) and Diver's Discount Network (DDN), with satisfactory performance.

For instance, in the following case, a cosmetic designer, ergonomic design specialist and a manufacturing engineer in Hong Kong have logged into the VRCE. A cosmetic designer might load a model of a new joystick design in the VRCE for others to


vnet (eai version) - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://iez059.ieem.ust.hk/
What's Related
cjsu
duffy
one
COSMO
Model URL:
Load
Users
Private Msg.
Switch View.
Log
Quit

Model URL: joystick_s.wrl

```
[duffy has connected.]
kan:  Hi everybody!
cjsu:  Hi
duffy:  Hi
kan:  Let's see my new design!
```

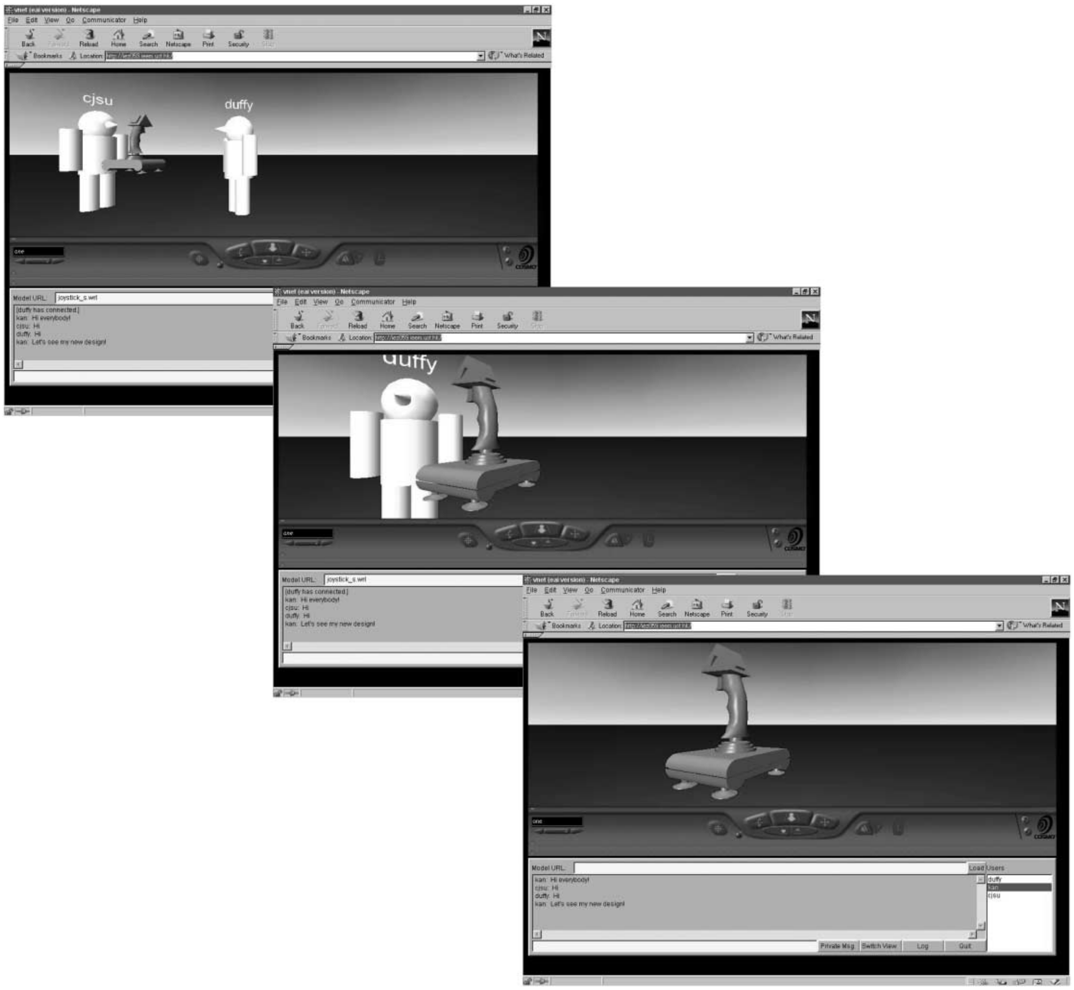
Users: kan, cjsu, duffy

Fig. 2. A shaded joystick model loaded by cosmetic designer in collaborative session.

comment. The cosmetic designer loads a shaded model as shown in Fig. 2. All collaborators are able to view the model in multiple viewpoints and switch to others' viewpoints with different perspectives supporting *Multi-view Collaborative Design Support*. For example, from the top snapshot to the middle snapshot of Fig. 3, the cosmetic designer changes his or her viewpoint to view the joystick model from a different view. The manufacturing engineer then switches his or her viewpoint to the same as the cosmetic designer as shown in the bottom snapshot of Fig. 3. They can also share ideas and experience with each other by text-based message. This supports the functional design concept *Multiple Opinions via Collaboration*.

The ergonomic designer might load a second layer of information, a wire-frame model of the joystick design in the VRCE (supporting the functional design concept *Multi-layer Information Exchange*). The wire-frame model allows the ergonomic designer to view the structure of the design and make some suggestions for the design.

For example, in a computer game joystick design process, a cosmetic designer would want to see the information as a three-dimensional structure, or as a shaded model, whereas a manufacturing engineer would want to see the same structure in terms of manufacturability, or possibly feasibility of die making. Providing multiple information layers over the


vnet (eai version) - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location What's Related
cjsu
duffy
Model URL:
Load
Users
Private Msg
Switch View
Log
Quit

Model URL: joystick_s.wrl

[duffy has connected.]
kan: Hi everybody!
cjsu: Hi
duffy: Hi
kan: Let's see my new design!

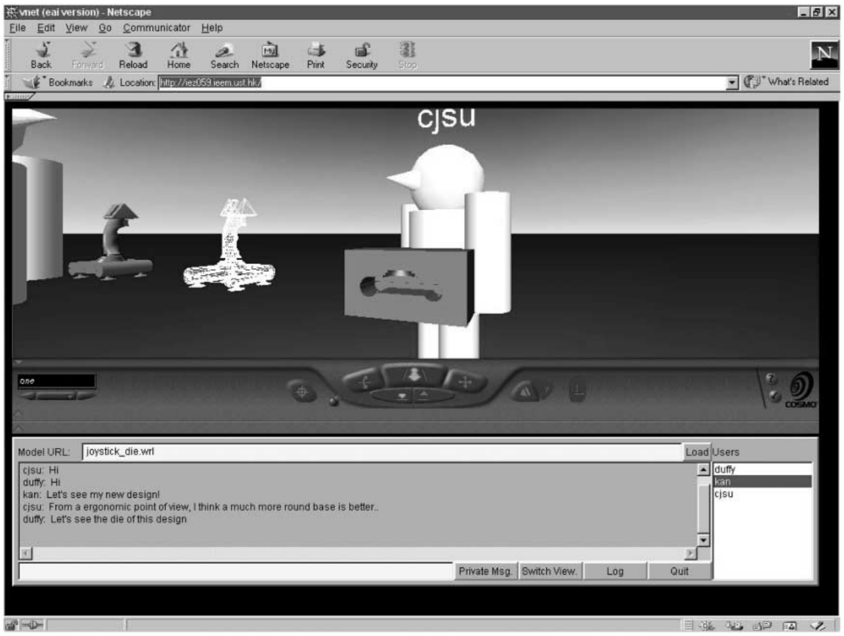
Model URL: joystick_s.wrl

[duffy has connected.]
kan: Hi everybody!
cjsu: Hi
duffy: Hi
kan: Let's see my new design!

kan: Hi everybody!
cjsu: Hi
duffy: Hi
kan: Let's see my new design!

Users: duffy, kan, cjsu

Fig. 3. Viewpoint changing and switching in VRCE.


vnet (eai version) - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://iez059.ieem.ust.hk/
What's Related
cjsu
one
COSMO
Model URL:
Load
Users
Private Msg.
Switch View.
Log
Quit

```
joystick_die.wrl
```

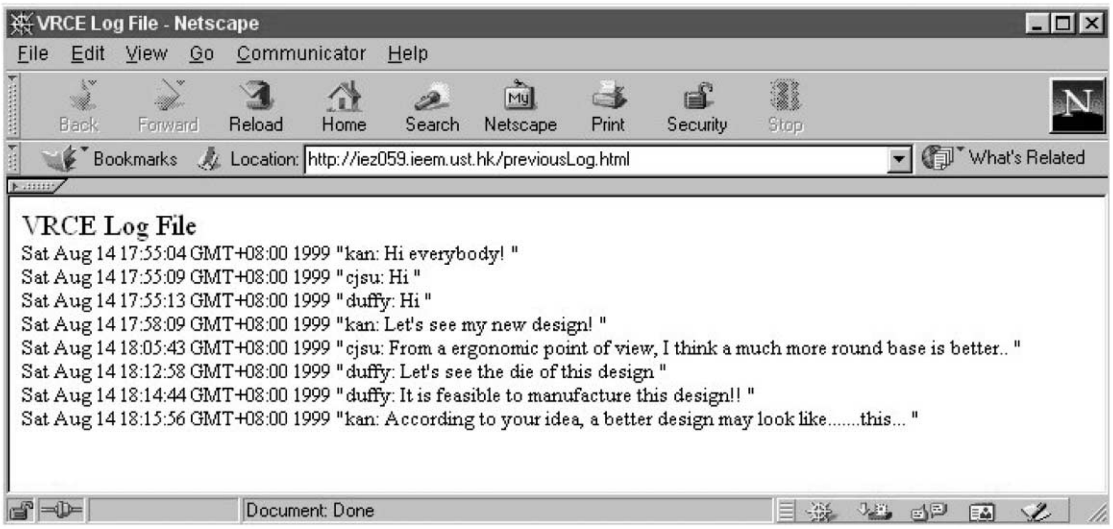
```
cjsu:  Hi
duffy:  Hi
kan:  Let's see my new design!
cjsu:  From a ergonomic point of view, I think a much more round base is better..
duffy:  Let's see the die of this design
```

```
duffy
kan
cjsu
```

Fig. 4. A wireframe joystick and a die model loaded in VRCE.

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.

The manufacturing engineer might indicate that the design from the cosmetic designer is feasible from the manufacturing point of view after loading a third layer of information. This information is a die model that


VRCE Log File - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://iez059.ieem.ust.hk/previousLog.html
What's Related
Document: Done

VRCE Log File

```
Sat Aug 14 17:55:04 GMT+08:00 1999 "kan: Hi everybody! "
Sat Aug 14 17:55:09 GMT+08:00 1999 "cjsu: Hi "
Sat Aug 14 17:55:13 GMT+08:00 1999 "duffy: Hi "
Sat Aug 14 17:58:09 GMT+08:00 1999 "kan: Let's see my new design! "
Sat Aug 14 18:05:43 GMT+08:00 1999 "cjsu: From a ergonomic point of view, I think a much more round base is better.. "
Sat Aug 14 18:12:58 GMT+08:00 1999 "duffy: Let's see the die of this design "
Sat Aug 14 18:14:44 GMT+08:00 1999 "duffy: It is feasible to manufacture this design!! "
Sat Aug 14 18:15:56 GMT+08:00 1999 "kan: According to your idea, a better design may look like.......this... "
```

Fig. 5. VRCE log file.

Table 3
Comparison between VRCE and three representative VCSs

	VRCE	dVISE	PIVOTAL	Deneb
Platform	Windows95, Windows98, Windows NT, Mac, UNIX, LINUX and IRIX	Microsoft Windows NT, HP UNIX and IRIX	Microsoft Windows and UNIX	Microsoft Windows NT and UNIX
Source code accessibility	Yes	No	No	No
Multi-view Collaborative Design Support	Yes	Partially	Partially	Partially
Multi-layer Information Exchange	Yes	Yes	Yes	Yes
Multiple Opinions via Collaboration	Yes	Yes	Yes	Yes
Maturing Design Ideas over Time	Yes	Yes	No	No
Experimenting with Multiple Designs	Yes	Yes	Yes	Yes
Secure Collaboration Support	Yes	No	No	No

could be used to mold the corresponding joystick shown in Fig. 4. After considering others' comments and ideas, the cosmetic designer might load another new joystick design into the VRCE. This supports the functional design concept *Experimenting with Multiple Designs*.

Later on, other virtual team members including the engineering manager, sales manager and marketing manager in the United States, with time difference, might join the virtual collaboration after the first group of collaborators have left. They then view the log to understand the progress of product design project, which supports the functional design concept *Maturing Design Ideas over Time* as shown in Fig. 5. They might load alternative models and continue the collaboration for the joystick design project in this continuously accessible virtual environment.

With this kind of virtual collaboration, a virtual prototype of newly designed joystick will allow an emerging agreement among collaborators with different expertise in a relatively short period of time. The design could then be passed to the production line for manufacturing with a reduced number of iterations in the product design life cycle. This can reduce time to market as well.

5.8. 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 produced by dVISE, PIVOTAL and Deneb. The comparisons are summarized in Table 3.

VRCE is equipped with all six functional design concepts. On the other hand, the other three representative 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 even similar security functions or concepts were not incorporated on any 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. Furthermore, portability and source code accessibility are two weaknesses of the three representative VCSs that were compared with VRCE.

6. Discussion

With the six functional design concepts, the VRCE has the potential to enhance efficiency and reduce iterations in the product design life cycle. One of the VRCE functional design concepts, *Multi-view Collaborative Design Support*, is useful for accurate space positioning in product design processes. It is considered to be useful and easily implemented even in a complex product design such as automobile body design. With the help of *Multi-view Collaborative*

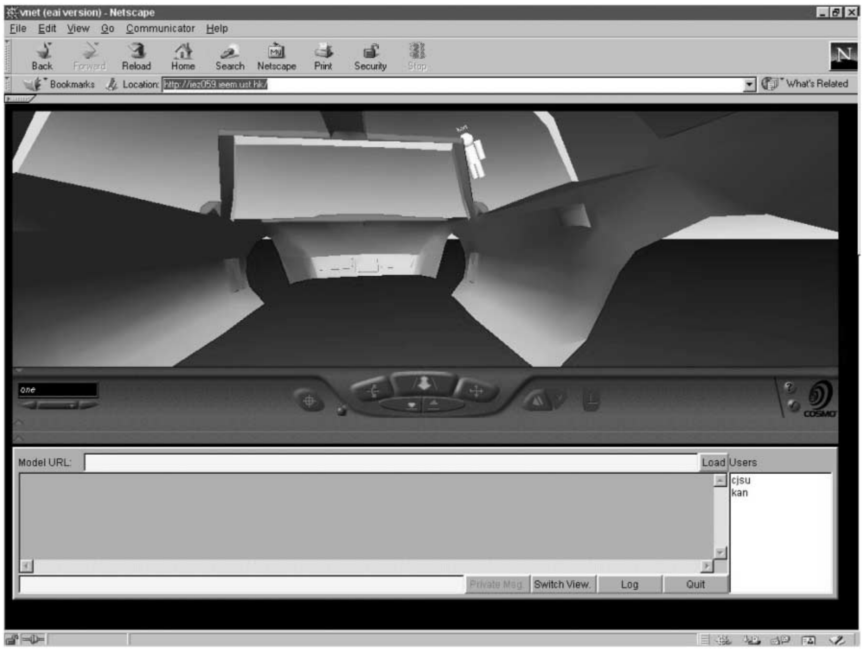

vnet (eai version) - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://iez059.ieem.ust.hk/
What's Related
Model URL:
Load
Users
cjsu
kan
Private Msg
Switch View.
Log
Quit

Fig. 6. The ‘‘inside-out’’ view of automobile body design in VRCE.

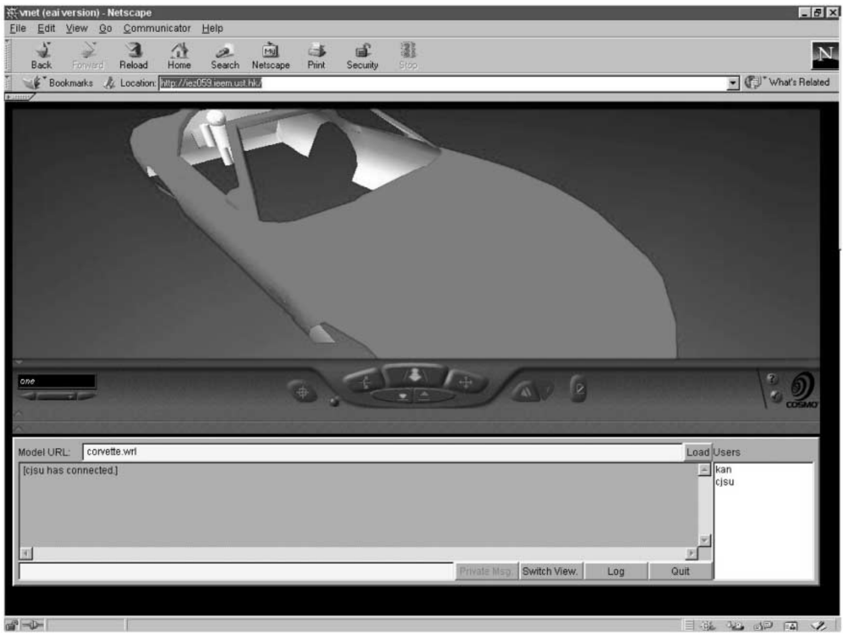
Design Support, automobile body designers and engineers are able to explore the design through different views. The ‘‘inside-out’’ view, as shown in Fig. 6, is useful for inner body design of an automobile whereas ‘‘outside-in’’ view, as shown in Fig. 7, is useful for outer body design of an automobile.

In order to utilize VRCE more fully, it is necessary to consider other VRCE functional design concepts. For instance, as stated previously, the *Multi-view Collaborative Design Support* within VRCE can be useful, and can be embedded easily for automobile body design. Yet there is a demanding knowledge requirement for embedding another functional design concept, *Multilayer Information Exchange*, in VRCE for automobile body design. As a result, the time for system customization increases. This is obvious when one considers requirements for embedding a Computational Fluid Dynamic (CFD) information layer into the existing VRCE for automobile body design. CFD contains complex data that most system developers would find difficulties incorporating. Thus, before considering use of VRCE, a 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 potential pitfall.

A conceptual model which is called 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 Fig. 8. The input of this model is the product complexity (z). The outputs of this model are the required knowledge for the embedding function in a 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. This implies that for a project with high product complexity, one would expect higher knowledge requirement for the VCS, and a longer time for VCS customization.

For instance, a joystick design, as illustrated, involves a relatively low product complexity. Thus, it is expected that a relatively low demand on required knowledge for the VCS would accompany a relatively shorter time for VCS customization for a joystick design. This is shown in the small cube in Fig. 9.

On the other hand, an automobile body design as illustrated in the last section involves a relative high


vnet (eai version) - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: What's Related
one
COSMO
Load
Users
Private Msg. Switch View. Log Quit

Location: http://iez059.ieem.ust.hk/

Model URL: corvette.wrl

[cjsu has connected.]

kan
cjsu

Fig. 7. The "outside-in" view of automobile body design in VRCE.

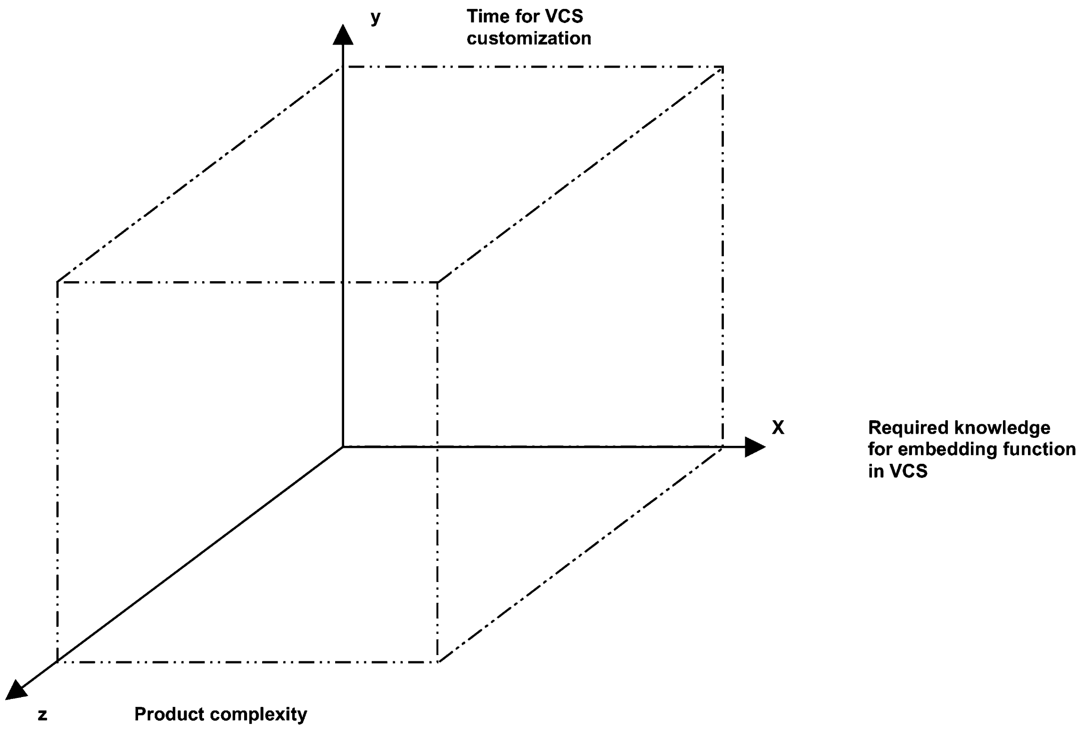

y
Time for VCS customization
X
Required knowledge for embedding function in VCS
z
Product complexity

Fig. 8. Graphical representation of VCS customization model.

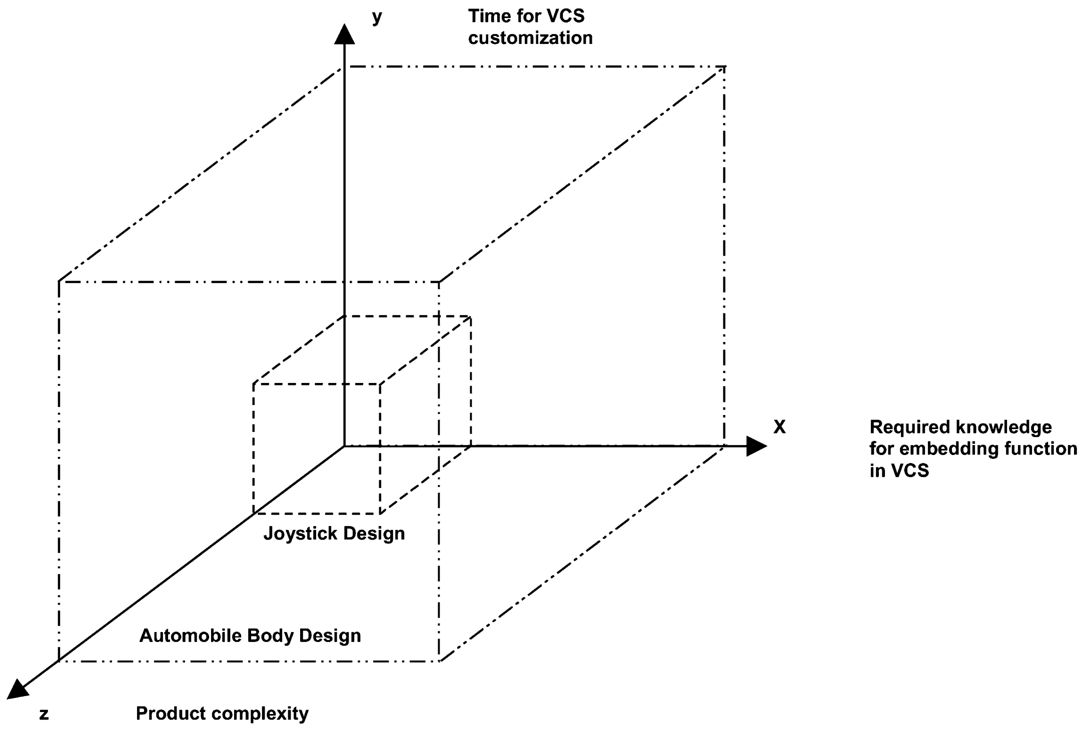

y
Time for VCS customization
X
Required knowledge for embedding function in VCS
Joystick Design
Automobile Body Design
z
Product complexity

Fig. 9. Joystick and automobile body design examples based on the conceptual model.

product complexity. Thus, it is expected that a relative large demand for required knowledge (for the embedding function in the VCS) would accompany a relatively longer time for VCS customization for an automobile body design. This is shown in the large cube in Fig. 9. This long system customization time has a negative impact on the potential benefits of VCS. Therefore, some assessment of these requirements should be made in advance in order to better predict whether use of VRCE would result in a reduction of total product development time.

7. Conclusion

VRCE was designed and developed to shorten the entire product design life cycle for teams in small to medium-scale industries that need to design products collaboratively from geographically distributed locations. VRCE is composed of a set of comprehensive functionality with customisability, is platform independent in nature and can be run using inexpensive computer hardware and software. The comprehensive functionality of VRCE includes six functional design concepts that are critical for effective product design in a collaborative virtual environment. Customisability makes VRCE more attractive than many existing VCS systems. Industries can make use of the platform independent VRCE without the high cost of changing operating systems. VRCE can reduce the investment cost for virtual collaboration. The feasibility of incorporating the necessary functional design concepts and system architecture was shown through development and testing of the VRCE prototype. A potential pitfall for using VRCE in certain circumstances was discussed and was illustrated with a proposed conceptual model.

References

[1] F. Dai, Virtual Reality for Industrial Applications, Berlin, Springer, 1998.

[2] K. Böhm, H. Wirth, J. Werner, Prototype Validation in Virtual Environments, Vision and First Implementation, [www]http://www.igd.fhg.de/www/gris/muse/publications/ifip/ifip.html (1999).
[3] N.E. Karangelen, N.D.T. Hoang, Simulation-based design for large complex computer-based systems, in: Proceedings of the Systems Engineering of Computer-Based Systems Workshop, IEEE Computer Society Press, Los Alamitos, CA, 24–27 May 1994, pp. 116–123.
[4] S. Aukstakalnis, D. Blatner, Silicon Mirage; The Art and Science of Virtual Reality, Peachpit Press, Berkeley, CA, 1992.
[5] J. Leigh, A.E. Johnson, C.A. Vasilakis, T.A. DeFanti, Multi-perspective Collaborative Design in Persistent Networked Virtual Environments, Project Report, University of Illinois at Chicago, 1998.
[6] Institute of Electrical and Electronics Engineers, Standard for Information Technology, Protocols for Distributed Interactive Simulation, International Standard ANSI/IEEE Std 1278–1993.
[7] J. Locke, An Introduction to the Internet Networking Environment and SIMNETIDIS, [www]http://www.npsnet.nps.navy.mil/publications/DISIntro.ps.Z (1999).
[8] W. Broll, DWTP — an Internet protocol for shared virtual environment, in: Proceedings of the Annual Symposium on the Virtual Reality Modeling Language, VRML 1998, ACM, New York, 16–19 February 1998, pp. 49–56.
[9] G. Bell, A. Parisi, M. Pesce, The Virtual Reality Modeling Language, Version 1.0 Specification, [www]http://www.vrml.org/technicalinfo/specifications/vrml1.0.htm (1999).
[10] J. Vacca, VRML Clearly Explained, 2nd Edition, AP Professional, Boston, 1998.
[11] C. Marrin, Proposal for a VRML 2.0 Informative Annex, External Authoring Interface Reference, [www]http://www.vrml.org/WorkingGroups/vrml-eai/ExternalInterface.html (1999).
[12] dVISE, [www]http://www.division.com/products/products.htm (1999).
[13] PIVOTAL, [www]http://www.centricsoftware.com/html/news/index.html (1999).
[14] Deneb, [www]http://www.deneb.com (1999).
[15] J.P. Harrison, Virtual collaborative simulation environment for integrated product and process development, IEEE International Symposium on High Performance Distributed Computing, Proceedings 1996, 6–9 August 1996, pp. 19–22.
[16] J.C. Westland, T.H.K. Clark, Global Electronic Commerce, MIT Press, Boston, 2000.
[17] E.R. Harold, Java Network Programming, 1st Edition, O'Reilly, Cambridge, 1997.
[18] T.A. Funkhouser, RING: a client-server system for multi-user virtual environments, in: Proceedings of the Symposium on Interactive 3D Graphics 1995, ACM, New York, USA, 9–12 April 1995, pp. 85–92.
[19] J. Weber, Using Java 1.2, 4th Edition, Indianapolis, Que., 1998.
[20] S.F. White, VNet, [www]http://www.csclub.uwaterloo.ca/~sfwhite/vnet/ (1999).
[21] M. McCarthy, A. Descartes, Reality Architecture, Prentice Hall, New York, 1998.
[22] S.F. White, VRML Interchange Protocol — Specification, [www]http://www.csclub.uwaterloo.ca/~sfwhite/vnet/VIP.html (1999).
[23] F. Lin, C.J. Su, M.M. Tseng, An agent based approach to developing intelligent virtual reality-based training systems, in: Proceedings of the 11th International Conference on Tools with Artificial Intelligence, Los Alamitos, CA, IEEE Computer Society, 1999, pp. 253–260.
[24] L.L. Ye, V.G. Duffy, B.P.-C. Yen, F. Lin, C.J. Su, Knowledge modelling methodology for intelligent virtual reality-based industrial training systems, Asian Journal of Ergonomics (2000), in press.
[25] N.H.M. Caldwell, P.J. Clarkson, P.A. Rodgers, A.P. Huxor, Web-based knowledge management for distributed design, in: Proceedings of the IEEE Intelligent Systems, IEEE, May–June 2000, pp. 40–47.

Ho-Yin Kan received his MPhil (Masters) degree in the Department of Industrial Engineering and Engineering Management at The Hong Kong University of Science and Technology in 1999. He continues to work on Internet and virtual reality software development projects throughout China and Hong Kong.

Vincent G. Duffy currently serves as an Assistant Professor in the Department of Industrial Engineering and Engineering Management at The Hong Kong University of Science and Technology. He was a Visiting Assistant Professor in the Industrial Engineering Department at the University of Wisconsin-Madison during part of 1999 and 2000, and received his PhD from Purdue University in 1996.

Chuan-Jun Su has been Assistant Professor in the Department of Industrial Engineering and Engineering Management at The Hong Kong University of Science and Technology and is founder of Horky.com Corporation in Taiwan. He received his PhD in Industrial Engineering from Texas A&M in College Station, TX.

View publication stats