Design and Implement Collaborative Virtual Environment System for Learning Butterfly Lifecycle

Lieu-Hen Chen
Department of Information Engineering, National Chi Nan University
1 University Road, Puli, Taiwan, ROC
lhchen@csie.ncnu.edu.tw

Cheng-Tin Lin
Department of Information Engineering, National Chi Nan University
1 University Road, Puli, Taiwan, ROC

Jen-Shin Hong
Department of Information Engineering, National Chi Nan University
1 University Road, Puli, Taiwan, ROC
jshong@csie.ncnu.edu.tw

Abstract

VRML—the Virtual Reality Modeling Language—is a description language for 3D worlds. The VRML supports sophisticated multimedia presentations and user interactions which has become the most popular 3D-object description language on Internet. However, current VRML technology only supports single-user navigation and interaction. The lack of a "collaborative virtual environments (CVEs) standard" in VRML has hindered the development of VRML-based multi-user application.

In this paper, we present a VRML-based "shared collaborative virtual environments" in an attempt to construct a multi-user 3D virtual butterfly ecosystem on the Internet for users to learn Taiwan butterflies. This prototype system integrates Java and VRML technologies to achieve the goal and proposes a solution to the implementation of interactive CVE systems.

Keywords : CVE, Virtual Reality Modeling Language (VRML), Multi-user VR, Distance Learning.

1 Introduction

The Virtual Reality Modeling Language (VRML)[8] is a file format for describing interactive 3D objects and worlds. VRML is designed to be used on the Internet, Intranets, and local client systems. Users can use VRML to create 3D objects and compose immersive and interactive virtual worlds. It is also possible to distribute a virtual world over the World Wide Web and let other people download and view it. While the first version of VRML [1] was a static scene description language, its second version [2] provide the possibility to create interactive virtual worlds.

However, the current VRML standard does not offer languages constructs for direct multi-user support. Most VRML scenes run on a single machine have limited interaction and only respond to a single user's input. Although the VRML community has already started to design and standardized it, the specification is still not released yet.

In recent years, many researchers have implemented many multi-user collaborative virtual environment systems [5][9]. These CVE systems have been successfully applied to chat tools [4]. Most of the CVE system developed so far are limited to certain specific platform, proprietary applications, or network protocols, and are not populated for the remote users.

In this paper, we have developed a virtual environment system to educate users about the lifecycle of butterfly. This system is implemented by integrating the VRML and JAVA technologies on a simple network communication mechanism.

This paper is organized as follows. Our prototype system and the designing considerations are introduced in section 2. Our approach to represent users and application in shared and collaborative VRML worlds are explained in section 3. The conclusion and future work is discussed in the final section.

2 System Architecture / Designing Consideration

One of the essential goals of CVE is to provide the capability of combining multi-participants and the information that they access and manipulate in a single place. We focus our designing consideration and research works on the following five components of a CVE system:

1. virtual worlds representation,
2. avatars,
3. shared objects,
4. dynamic control,
5. network communication.

To achieve its essential goal, our CVE system consists of the above five components. Each component plays a basic and important role. We discuss and explain them separately in the following sections.

2.1 Virtual Worlds Representation

First of all, a CVE system has to provide a shared environment for users to cooperate with each other. For our system, this environment is a 3D virtual world shared by multi-users over the Internet. As mentioned before, VRML is a very good solution for constructing, distributing, and rendering such kind of 3D worlds over the Internet. In addition to the above advantages, VRML provide programmers a variety of nodes such as the interaction, animation, and sensor nodes to serve different purpose. With the combination of nodes, we are able to create real and nature behaviors of objects easily in a VRML world as shown in Figure 1.

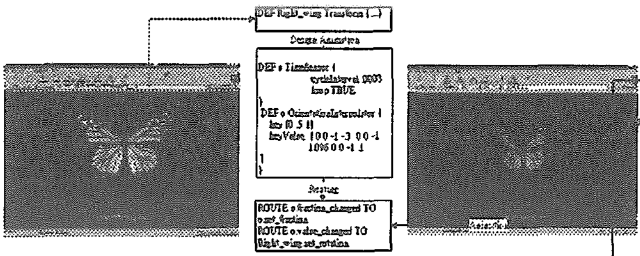

Figure 1 Design butterfly flying animation with VRML

2.2 Avatars

After the virtual world has been prepared, our CVE system is ready for its participants to enter.

The avatars are the virtual objects representing the participants in the virtual world. A good avatar node has the following features [5]:

1. Avatar specific behaviors.
2. The relative position and orientation.
3. Interaction parameters.
4. Identified name.

Unfortunately, current VRML specification does not provide avatar node directly. Programmers must define their own avatar nodes by themselves. Our application focused on learning the butterfly lifecycle. Thus, we design and implement the butterfly avatar node represented the participants in our virtual ecosystem. The children of the butterfly node defines the actual appearance of the avatar as well as the avatar dependent behavior (e.g. a flying animation to represent the fly-through movement of a butterfly avatar). Participants must control their own avatar directly by browser (Figure 2). The VRML definition of our butterfly avatar node is shown in Figure 3.

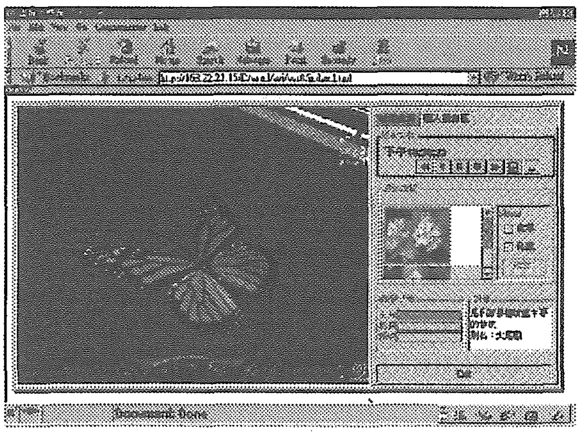

Figure 2 Butterfly Avatar

Basically, each VRML file is downloaded from server to client only once. In order to retrieve and update each avatar's information dynamically and efficiently, we need to introduce a network communication architecture into VRML to solve the above problem. When each participant login to our system, it is assigned an avatar node with an unique identifier. By taking advantage of this user management strategy and introducing Java's client-server architecture, the Remote Method Invoke (RMI) technique, into our system, we achieve a solution to the above problem. More detail about networking is discussed lately in section 2.5.

```
PROTO Butterfly
[
    exposedField MFString name              [""]
    eventIn MFString url

    exposedField SFVec3f alltranslation     0 0 0 1
    exposedField SFRotation allrotation     0 0 0 1
    exposedField SFVec3f translation        0 0 0
    exposedField SFRotation rotation        0 0 0 1

    exposedField SFBool view_enabled        TRUE
    exposedField SFBool avatar_enabled      FALSE
    exposedField SFBool drag_enabled        TRUE

    exposedField SFBool touch_enabled       FALSE

    exposedField SFVec3f scale        1 1 1
    exposedField MFNode children      []
    exposedField MFNode allchildren   []
    eventIn MFNode addChildren
    eventIn MFNode removeChildren
    eventIn SFBool set_bind

    eventOut SFBool isActive
    eventOut SFVec3f translation_changed
]
{........}
```

Figure 3 Butterfly Proto-Node

2.3 Shared Objects

A complete collaborative virtual environment consists of avatars and other objects. There are various kinds of interaction among objects and participants in a virtual world. Sometime, one simple interaction between an avatar and an object may cause very complicate chain-reactions in the world. Therefore, it is very important to manage these objects concurrently, instantaneously and correctly in an efficient manner. It is necessary for program at server site to manage each shared object, store their states, and authorize their control rights to clients. In our system, shared objects are defined in a similar way as the butterfly avatar node is.

2.4 Dynamic Control

For a local VRML world to communicate with external environment, an interface is necessary. This interface is called an External Authoring Interface (EAI). EAI defines the set of functionality of a VRML browser that the external environment can access. [3][6] This interface is designed to allow an external environment to access nodes in a VRML scene using the existing VRML event model.

EAI Specification was developed with Java. Java applet is used to manage and control the named nodes in the virtual environment. EAI event architecture is shown in Figure 4.

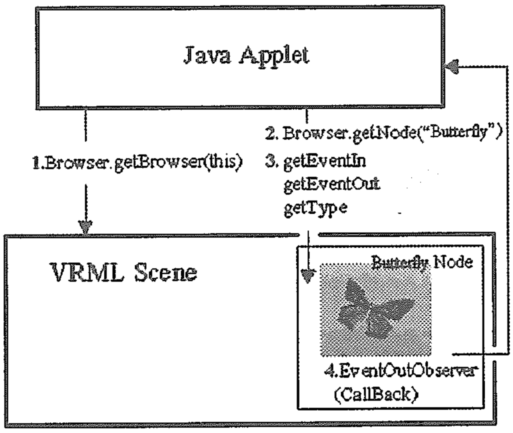

Figure 4 EAI events architecture

2.5 Network Communication

Because VRML downloads the whole files to client site through Internet only once at the beginning, it is very difficult for server site to handle local situation at each client site. Therefore, a client-server architecture that allows message exchanging between client and server is necessary. Fortunately, Java specification has already supported such programs. A new mechanism called Remote Method Invoke (RMI) can facilitate us to do this task [7]. Figure 5 shows the Java RMI mechanisms.

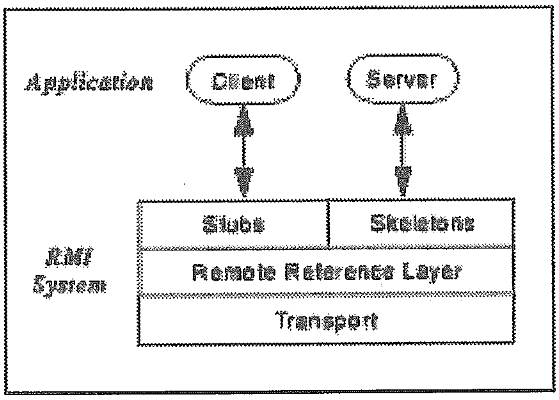

Figure 5 Java RMI Mechanism

There are various types of messages need to be transmitted in a CVE system. (E.g. MFString, SFVec3f, SFRotation...etc). In order to determine sender, receiver, and the content of the message, we need a new class to handle these complex messages. We defined a class, "DataSet", to store different message types. Figure 6 shows our CVE system message types and DataSet class.

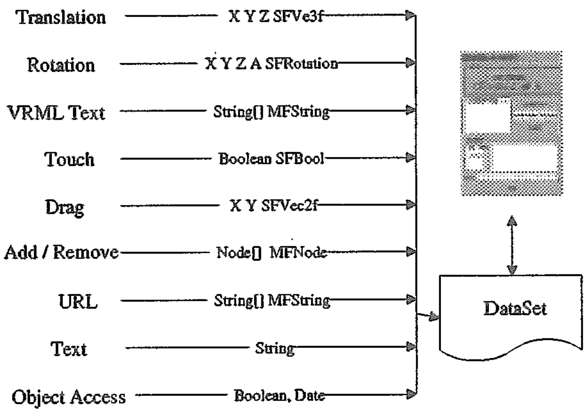

Figure 6. Message types

Each client sends messages to server. Server site receives, stores and broadcasts these messages for every participants.

As shown in Figure 7, there are three sections in server site: Web server, User-profile Server, and CVE server in the network architecture of our prototype system. There are two sections in client site: Single-user and Multi-user. Web-server keeps our system and maintains network linkages to other machines. User-profile server takes charge of users management including add and delete users, record user profile, etc.. CVE server handles the event message exchanging among all clients. There are two databases in server site. One is for VRML models and the other is for user information. Shared objects, avatar models and user identification information are all kept here.

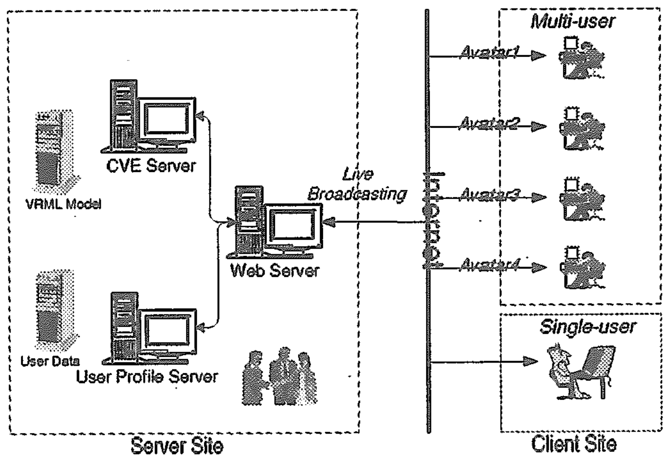

Figure 7 System architecture

In order to simplify the system designing consideration and improve the performance according to different user's requirements, we design two client mode: single-user and multi-user section. Single-user system can facilitate users to adopt virtual butterflies, feed them, and learn the lifecycle of butterfly. The feeding results will be record by system as part of user's profile. Multi-user system provides a collaborative virtual environment, a virtual world. As their avatar in this cyberspace, each participant can exhibit their own butterfly, which is brought up and recorded in the single-user section. They can fly-through and compete for resources in this environment. They can also chat to each other and exchange their experience.

3 Implementation of CVE System

As mentioned before, the server site consists of web server, user-profile server, and CVE server. On the other hand, the applications at client site are divided into two different category, single-user and multi-user sections, according to user needs. The RMI mechanism offered by JDK enables us to design and implement client-server architecture programs to achieve the above goals.

3.1 Server Site

A CVE server interface is shown in Figure 8.

This interface provides 11 methods for client programs to invoke. Through this interface, system programs at server site handle events transmitting and shared objects management, monitor online user's situation, assign every user a unique avatar name, etc..

All information including behaviors of avatar, shared objects access control, and text messages must be stored in the server site and broadcast to every online user. The copy of virtual world at client site will then be updated. Figure 9 shows the server management items.

```
public String login(String id) throws RemoteException;
public int usernum(String id) throws RemoteException;
public void postMsg(String id,DataSet msg)
        throws RemoteException;
public void exit(String id,DataSet name)
        throws RemoteException;
public String[] loaduserList() throws RemoteException;
public Vector getNewMsg(String id)
        throws RemoteException;
public DataSet loadExistNode(String id)
        throws RemoteException;
public void postNewUser(String id, DataSet user)
        throws RemoteException;
public Date getCurrentDate() throws RemoteException;
public boolean queryShareObject(String id,
        String shareobject) throws RemoteException;
public void setShareObjectAccess(String shareobject,
        DataSet shareobject) throws RemoteException;
```

Figure 8 CVEs-server interface

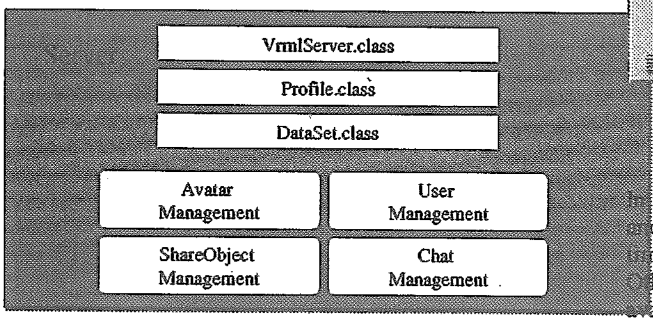

Figure 9 Server management items

3.2 Client Site

We use Java swing to design and implement the user interface at client site. As shown in Figure 10, we divide this interface into three components: VRML browser, Java applet and HTML component. The outlook of this interface is shown in Figure 11 and Figure 12.

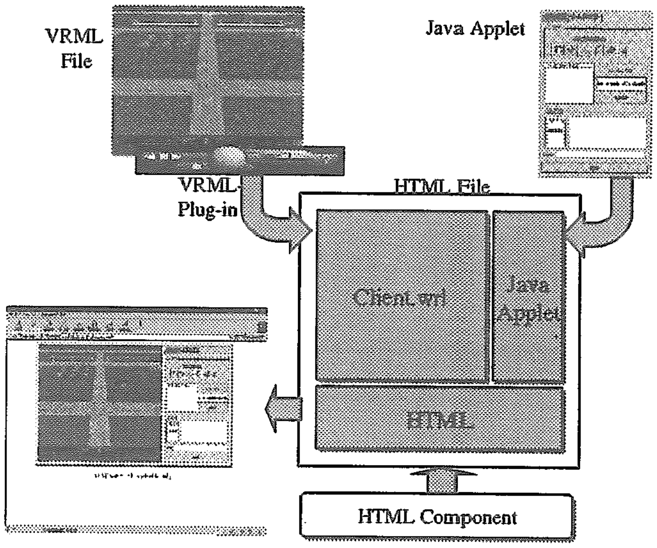

Figure 10 Components of user interface at client site

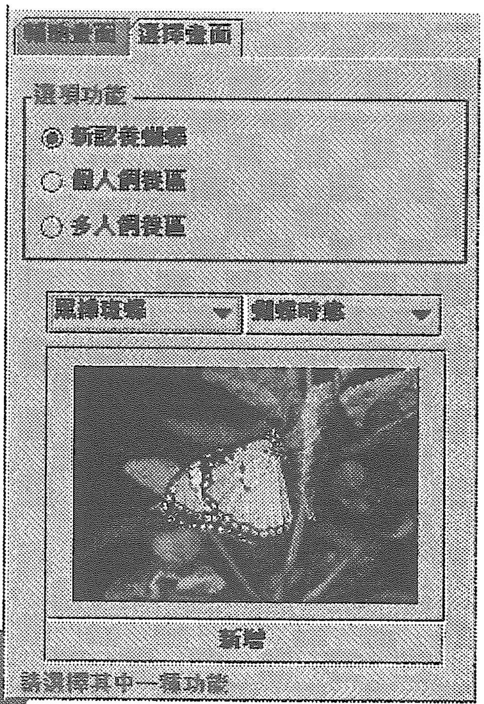

Figure 11 example of JAVA applet user interface

In the example of Figure 12, there are two users, nickey and ppp, participate in the virtual environment. The system time shown on tool list is actually the time at server site. Other buttons on tool list allow user to control their own avatar in the virtual world.

Shared objects list prompts the identification name of shared objects in the virtual world to users. User can try to gain the control of these objects by directly selecting their names on the list. When this selection is a legal action, the button on tool list will be active.

Typing the full address at the URL field will make system to load the use-defined model into the virtual world.

Text chatting field serves the role of chat-room. It allows users to exchange their knowledge and experience.

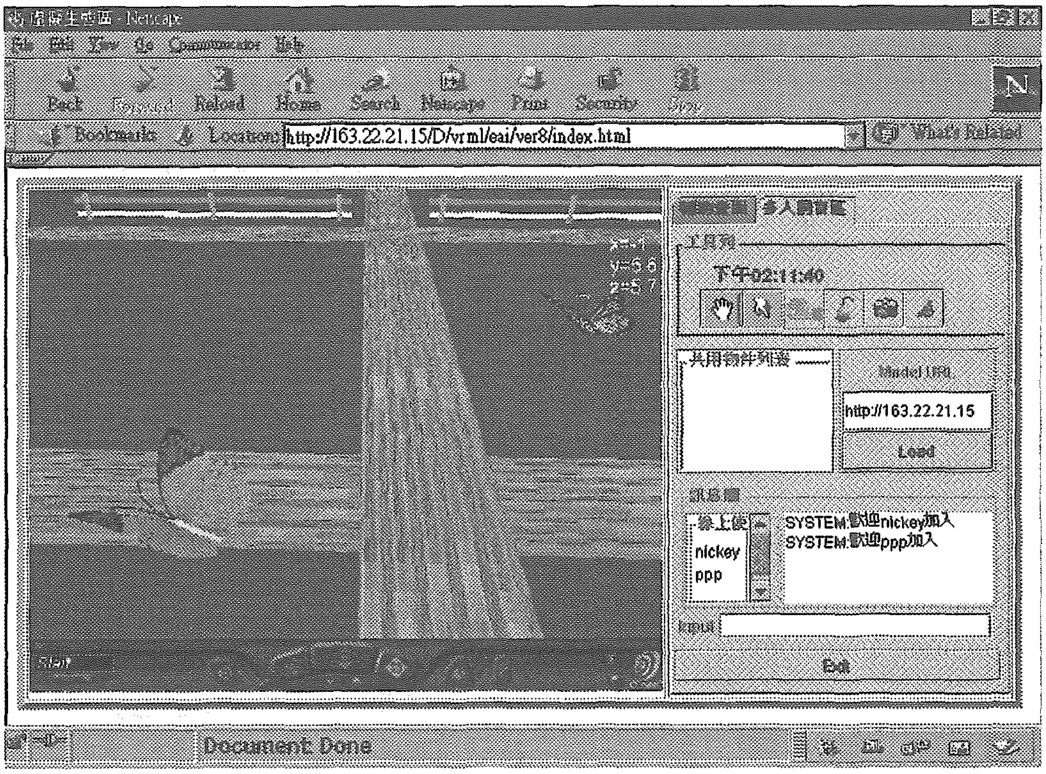

Figure 12 CVE system client interface (1)

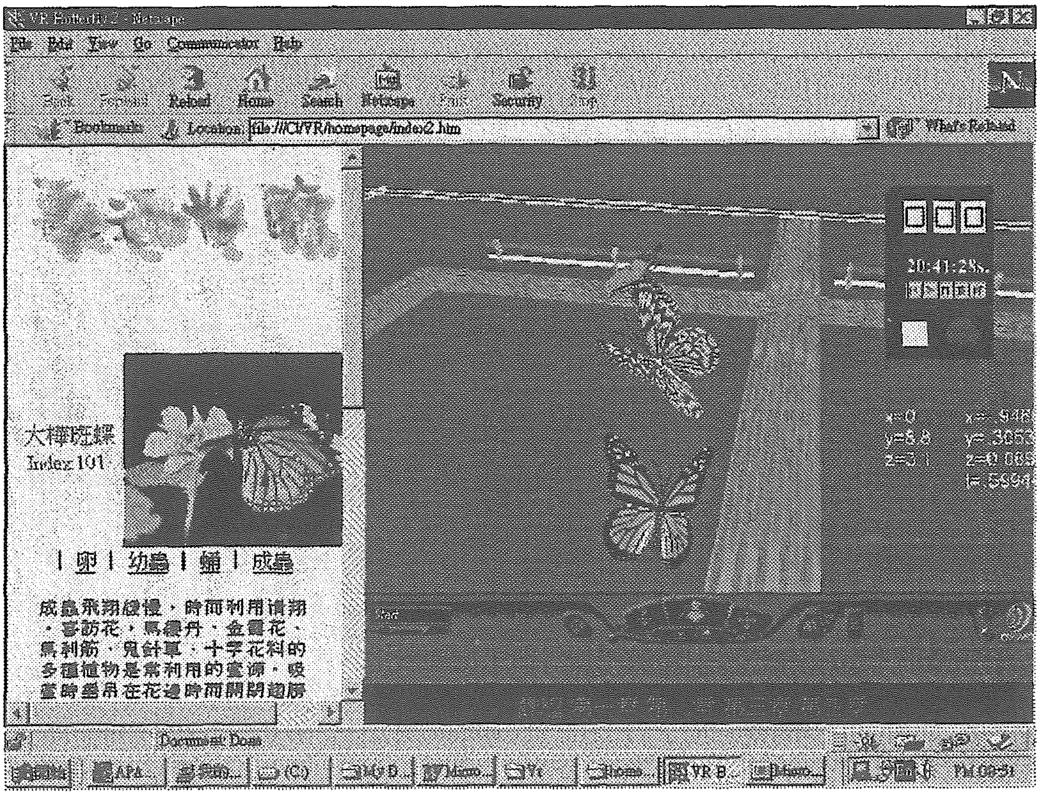

Figure 13 CVE system client interface (2)

Another example of our CVE system is shown in figure 13. Each user can breed butterflies as her / his own virtual pets in the system. All these information is kept in the user profiles. When users login into the multi-users section, they will be asked to choose one of the butterflies from the profiles to serve as an unique avatar.

4 Future Work and Conclusion

We have presented a Collaborative Virtual Environment system which is developed for learning butterfly lifecycle. It integrates Java and VRML technologies to provide an interactive multi-users virtual ecosystem on the Internet. The proposed prototype system successfully overcomes the current VRML disadvantages on CVE development. In addition, the proposed system provides an environment which allows the users to participate altogether on the Internet in discussing and learning the lifecycle of butterfly. The average performance of system is about 10 frames/sec when there are four users in the system.

There are various problems remain to be solved in the VRML. One of them is the expensive downloading and rendering cost caused by the file size. VRML users have to download the whole 3D scene before starting navigation. Due to the vast data amount of typical 3D objects, the downloading time is intolerable in a low-bandwidth network environment. More research efforts will be focused on the topics of automatic model-simplification and dynamic Level of Detail to solve the above problems.

5 Acknowledgments

This work is supported by NSC-88-2745-P-260-006.

References

[1] A. L. Ames, D. R. Nadeau, and J. L. Moreland. "The VRML Sourcebook", John Wiely, New York,1996.

[2] A. L. Ames, D. R. Nadeau, and J. L. Moreland. "The VRML2.0 Sourcebook", John Wiely, New York, 1996.

[3] D. Brutzman, "The virtual reality modeling language and Java", Communication of the ACM, pp. 57—64, Jun. 1998.

[4] E. Anneli, A. Nou, "Building Distributed Virtual Environments to Support Collaborative Work.", VRST'98 , pp. 105—114, Nov. 1998

[5] W. Broll, "Populating the Internet: Supporting Multiple Users and Shared Application with VRML", Proceedings of the second symposium on Virtual reality modeling language, ACM, pp. 33—40, 1997.

[6] Proposal for a VRML 2.0 Informative Annex External Authoring Interface Reference, http://www.cosmosoftware.com/developer/moving-worlds/spec/ExternalInterface.html .

[7] RMI Documentation, http://java.sun.com/products/jdk/1.1/docs/guide/rmi/index.html .

[8] VRML 97--ISO/IEC 14772 Index Page, http://www.web3d.org/WorkingGroups/vrml-eai/Specification/ .

[9] 陳恆佑, 林建廷, 秦其巍, 陳景宜, 洪政欣, "『虛擬黑板』: VRML 在遠距教學之應用", The 8th International Conference on Computer-Assisted Instruction, ICCAI, 1999