Application of Agent-aided CVE in Remote Collaborative Experiment

Yong Liu[1], Xiaohong Mi[1], Shaozi Li[2]
1. College of Electronic Information Engineering, Henan University of Science & Technology, Luoyang 471003, China
2. Cognitive Science Department, Xiamen University, Xiamen 361005, China
LuoYangLiuYong@163.com

Abstract

Since experiment is one collaborative activity and usually needs people collaborate together, it's very important to study how to realize the collaboration among users in remote experiment. In this paper, the architecture of an Agent-aided cooperative model and its cooperation mechanism are proposed. Based on the architecture, one Agent-aided collaborative virtual experiment system is built. It proves that by applying CVE and Agent technologies we can provide an effective way to manage the remote collaborative experiment.

Keywords: Agent, Collaborative Virtual Environment, Remote Experiment, Cooperation Mechanism

1. Introduction

Along with the development of the distributed computer system and high-speed network, computer systems have been developed from the traditional problem solving, some scientific calculation to helping user interaction and supporting cooperative work [1]. Nowadays, it's possible for people to be educated through Internet. But, for those who have poor experimental condition, whether it's possible for them to make the experiment through Internet? And how do they cooperate with others in remote experiment?

In the last days, some experts have suggested building Network Virtual Lab, which uses VR-Virtual Reality technologies to simulate some instruments and phenomenon of the experiment. This virtual lab provides possibilities for people to make the experiment and watch its phenomenon through Internet. By now, many institutes oversea have succeeded in building their virtual labs, such as Carnegie Mellon's Virtual Lab of Carnegie-Mellon University, Virtual Engineering/Science Laboratory of Johns Hopkins University, and Engineering Laboratories on the Web of the University of Tennessee at Chattanooga and so on.

But most of these studies mainly focused on the building of the virtual environment of the lab, few considers about the collaboration among users located in different places. We all know that experiment is one collaborative activity and usually needs people collaborate together. So, it's very important to study how to realize the collaboration among users in remote experiment. This paper proposes to apply Agent-aided CVE to solve this problem.

CVE-Collaborative Virtual Environments is shared virtual reality spaces where remote users participate in a simulation. In addition, CVE also allow the participants to collaborate in closely coupled and highly synchronized tasks. These tasks require very close coordination between two or more users, such as two remote surgeons operating on a remote virtual patient. The applications of such environments are in military training, collaborative design and engineering, distance training, entertainment, and many other personal and industrial applications [2].

Agents, proactive and adaptive entities, can provide solutions to some of the problems existing in CVE. They can solve the following problems proposed in [3] due to their characteristics of autonomy, intelligence and ability to learn: how participants should be represented in collaborative virtual environments; how participants should interact with each other in a shared space; how to filter relevant information to reduce processing at each client for large worlds; how to sustain a virtual environment even when all its participants have left. Due to their powerful abilities, software agents have been focused in CVE for several years.

In order to realize the collaboration among users in the remote experiment, one Agent-aided cooperative model and its cooperation mechanism are being

978-1-4244-2511-2/08/$25.00 ©2008 IEEE

addressed. In this paper, we apply CVE and Agent technologies to build one agent-aided collaborative virtual experiment system. Therefore, the collaboration among users becomes more efficient and more intelligent.

2. System Architecture and Relevant Technologies

2.1 System Requirement

According to the definition of the CVE, we can see that the virtual environment of the remote collaborative experiment must meet the following needs:

(1) Simulates the experiment like the real way. The three-dimensional (3D) virtual environment should provides relevant instruments and be able to simulate phenomenon of the experiment to enhance the immersion of the users.

(2) Shares data among users. The data of the virtual environment should be shared by users in some way. If one of them manipulates the shared object, others should be able to receive the latest view. All of them should have the coordination view of the virtual environment.

(3) Provides cooperation mechanism. The user attended in the virtual experiment should be able to collaborate and interact with others, so they can achieve the remote collaborative experiment efficiently.

2.2 System Architecture

From the above points, we know that the remote collaborative experiment needs users located different places share the same view of the virtual experiment, so the system should be able to keep all the environment data sustained and deliver any change of it to clients in time. At the same time, the system should be able to deliver collaboration and interaction requirements from one user to another and apply relevant mechanism to help them cooperate with each other.

Based on this, we adopt client/server software architecture, which is a modular infrastructure to improve usability, flexibility, interoperability, and scalability. And we employ centralized data management with data replication on demand. A client knows at most one world at any moment. The centralized system can co-ordinate the activities of users.

The remote collaborative experiment system is client/server based. The server is responsible for managing users' connection and answering users' requirements. The client can access the server through Internet by using Java applet. The virtual experiment environment is edited in .wrl (VRML format) and is hosted on server. When the user logs to the server, he has the option to choose the experiment that he wants to make. After he selects, the relevant VRML files will be downloaded automatically from the server and the experiment environment will be displayed in the explorer after interpreted by plug-in. Based on the experiment the user joins, he is assigned an agent and his name is added in the agent's list. He can make the experiment with the help of the agent and collaborate with others through the cooperation mechanism.

2.3 Relative Technologies

(1) VRML

The Virtual Reality Modeling Language (VRML) is a file format for the description of dynamic scene graphs containing 3D objects with their visual appearance, multimedia content, an event model, and scripting capabilities. VRML is designed to be used on the Internet and local client systems and to be used as an exchange file format. VRML is the universal language for integrated 3D graphics and multimedia [4]. The multi-user distributed world use VRML as the rendering and interaction vehicle. By using JAVA as the scripting language and its network function, VRML can build one collaborative virtual lab.

(2) JAVA

Java is an object-oriented programming language developed by the Sun Microsystems. Java is a web programming language, which supports platform independence. Java code can be executed in HTML browsers and used as the scripting language in VRML as well. Java applets can open network connections to servers as well as present graphical user interface. Furthermore with the help of External Authoring Interface, it can access the VRML plug-in functionality [5] to display and control visual simulation.

(3) EAI

EAI-External Authoring Interface defines the interface that applications external to the VRML browser in our case the applet to access and manipulate the objects in the VRML scene. EAI connects the Java Virtual Machine running in the web browser to execute applets and plug-in used to display VRML content. It is accessed with a set of Java classes defined in the EAI Specification. The client uses the EAI to control the visual simulation in the VRML plug in. Thus EAI allows the external environment to access the nodes in the VRML scene using the existing VRML event model.

The External Authoring Interface allows four types of access into the VRML scene: (1) Accessing the functionality of the Browser script interface, (2) Sending events to eventIn of nodes inside the scene, (3) Reading the last value sent from event outs of nodes inside the scene, and (4) Getting notified when events change values of node fields inside the scene.

3. Application of Agent-aided CVE

3.1 Agent Model

In fact, an intelligent agent is a computer system capable of flexible autonomous action in some environment. The main features about agents are as follows [6]:

(1) Autonomy: capable of acting independently, exhibiting control over their internal state by flexible;

(2) Social Ability: the ability to interact with other agents (and possibly humans) via some kind of agent-communication language, and perhaps cooperate with others, and agents interact with environment through sensors and effectors as shown in Figure 1:

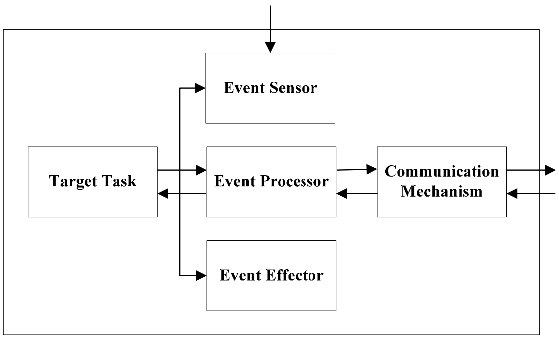

Figure1. Agent model

Event Processor is the heart, it can response in real time. The event sensor perceives certain event statuses and triggers different event adapter to work according to event type; the adapter matches the event and sends its result to event distributor to trigger correlated method to solve the event.

3.2 Architecture of the Agent-aided Cooperative Model

The architecture is divided into two logical aspects interconnected closely with each other. The first aspect is the information exchange and coordination between the system components and the agents. The second is the design and cooperation of the agents themselves. These agents are used to interact with the user, offer a homogeneous user interface and support the cooperation work between different users. The architecture of the Agent-aided cooperative model is shown as Fig.2. And the following is a description of its components.

(1) The Console Agent is the console component to control the whole collaborative virtual experiment system. It is mainly responsible for verifying registered users, allocating tasks and sustaining the environment, managing resource databases, detecting collisions and controlling co-manipulation of collaborative experiment.

(2) The Agent, a surrogate of a user in the system, provides the user with a homogeneous interface for registering into a CVE, tracing the user behaviors, communicating with the Console Agent and other agents.

(3) The Software Bus is in charge of the communication between Console Agent and other agents belonging to different users and communications between agents and resource database. The Software Bus can be HLA [7] or CORBA [8].

(4) The Resource Database consists of a serial of experiments and each experiment is kept in VRML files. When the user logs in the system, he can select the experiment he wants to make from the resource database.

(5) The Information Database is distributed on the network. They keep the information about users, such as experiment manipulation and communication with other users, as well as performing results obtained by agent for completing specific tasks.

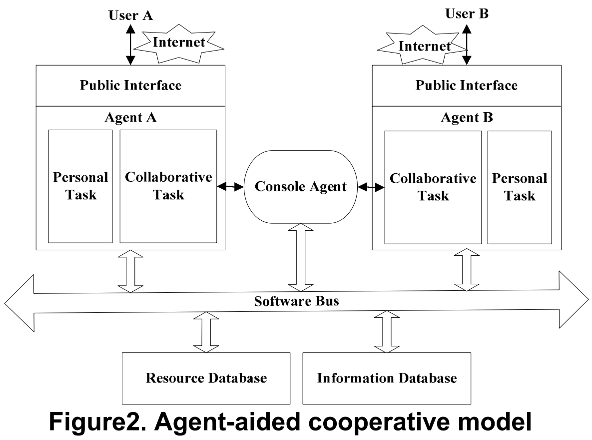

Figure2. Agent-aided cooperative model

3.3 Cooperation Mechanism

Cooperation mechanism is the kernel of the remote collaborative experiment. It is implemented by Console Agent to provide supports for users' Agent to make experiment together. And the Console Agent also provides services of virtual environment data access and coordination for remote experiment.

For one experiment, if there is only one user join in, he can make it by himself with the aid of his Agent. Otherwise, the Console Agent will dynamically organize cooperating process, manages shared cooperative experiments, and allocates the tasks, the control access rights over cooperative objects.

After the experimental tasks allocated, the cooperative members should wait for his turn to control the cooperative objects. When a user wants to gain the rights to control the cooperative objects, he must first makes a request to the Console Agent for the rights, after being granted, he can process some permitted operation. After operating, he will report to the Console Agent, and the Console Agent will broadcast the latest result during the cooperative processing. The cooperation mechanism is showed as Figure 3.

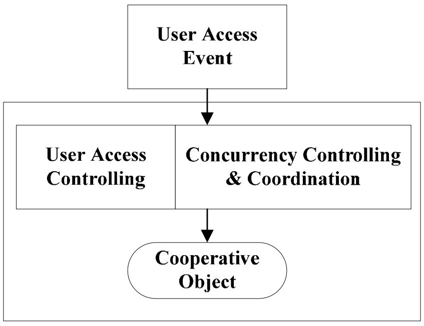

Figure3. Cooperative mechanism

In the process of cooperation, the Console Agent grants the rights to the cooperative member or ignores his requests as if he asks for the operating rights. Main logic, which exists in such mechanism, is:

(1) Console Agent organizes the cooperative experiment.

(2) Cooperative member delivers his request to the Console Agent for gain the operating rights.

(3) Console Agent answers the request (grant rights) or ignores the cooperative member's request according to the situation of the cooperative process.

(4) Console Agent takes the duty to refresh other member's display simultaneously after cooperative member's operating.

(5) Console Agent can deprive other member’s rights of making experiment on cooperative object at any time if the cooperative process need.

(6) Console Agent broadcasts all the information including the cooperative introduction, its results and so on.

3.4 Simulation Tests

Based on the above architecture, a demo system was designed and implemented. This demo system is about a virtual assembling experiment as shown in Figure 4. The Console Agent is used to manage the experimental environment, including registration of users and virtual stuff, updating changes in the environment, tracing users, and sustaining the whole system. As a result, when no users connect to the virtual environment, the server waits for a user coming; when users log in, they can make the experiment by himself with the aid of his Agent or cooperate with others.

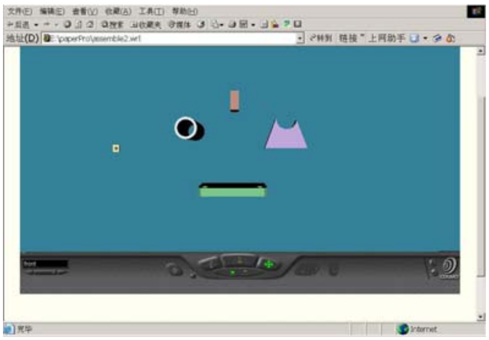

Figure 4(a). Initial assembling environment

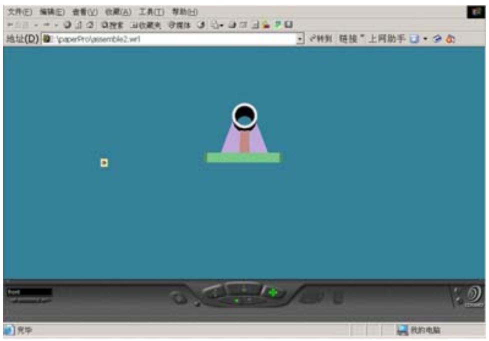

Figure 4(b). Environment after collaboration

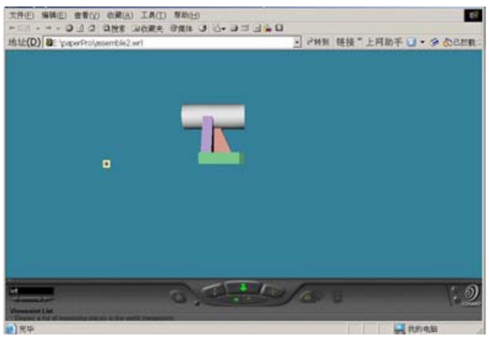

Figure 4(c). Different view of the environment

4. Conclusions

In this paper, the architecture of an Agent-aided cooperative model and cooperation mechanism are proposed and described in detail. As a demonstration of the proposed architecture, a simple demo system is presented and discussed. It proves that by building Agent-aided CVE we can provide an effective way to manage the remote collaborative experiment. However, the current prototype is not complete and there are still many problems to be solved. For example, since we apply Agent technologies in our system, we should consider about the security problem. And another one, perhaps we can also provide users with avatar, the representation of user in the virtual environment to enhance users' immersion. Both of these will be our future tasks.

5. References

[1] Gao Sheng and Wu Lin, "Collaborative mode for remote welding robotic system based on computer-supported cooperative work", Hanjie Xuebao/Transactions of the China Welding Institution, March 2007, v 28, n 3, pp. 9-12.

[2] Zheng Chen and Jianhua Gong, "Platform development of collaborative virtual environment based on Java3D", Application Research of Computers, 2005, Volume 22, pp.136-139.

[3] Y. Zhang, L. Guo and N.D. Georganas, "AGILE: An Architecture for Agent-Based Collaborative and Interactive Virtual Environments", Proc. IEEE Globecom 2000 Conference, San Francisco, 2000, pp.1-9.

[4] T.M Dave, Yanqing Zhang, G.S Owen and Raj Sunderraman, "Intelligent Web Agents for a 3D Virtual Community", International Journal for Infonomics, Issue 1, January 2005, pp.38- 49.

[5] Yu Chunyan, Wu Minghui and Wu Haihong, "Combining Java with VRML worlds for web-based collaborative virtual environment", ICNSC2005-2005 IEEE Networking, Sensing and Control, 2005, pp. 299-304.

[6] Wang Dong and Yu Tao, "Study on virtual intelligent assembly system for machine tools based on multi-agent and Petri-net", Proceedings of the 2006 International Conference on Machine Learning and Cybernetics, 2006, pp. 28-33

[7] http://hla.dmso.mil/tech/.

[8] http://www.omg.org/gettingstarted/corbafaq.html.