An Educational Community Using Collaborative Virtual Environments

Christos Bouras[1,2], Dimitrios Psaltoulis[1], Christos Psaroudis[1], and Thrasyvoulos Tsiatsos[1,2]

[1] Computer Engineering and Informatics Dept., Univ. of Patras, Greece
{bouras,psaltoul,psaroudi,tsiatsos}@ceid.upatras.gr
[2] Computer Technology Institute, Greece
{bouras,tsiatsos} @cti.gr

Abstract. The use of Collaborative Virtual Environments in e-learning is one of the most promising uses of the virtual reality technology. While a lot of research has been done in the area of collaborative virtual environments corresponding to the sharing of events, very little research has been done on specific services and functionality. However both the requirements and the kind of the offered services affect significantly the design of a system. In this paper we present an Educational Community to support e-learning services using Collaborative Virtual Environments from both the technical and functional point of view.

1 Introduction

On the past few years a number of interactive virtual reality (VR) systems have been developed. An Educational Virtual Environment (EVE) [1] is a special case of a VR system where the emphasis is more on the "education and collaboration" rather than on simulation. EVEs actually are Collaborative Virtual Environments (CVEs) [2] that can be used for educational applications such as collaborative e-learning. Collaborative e-learning is any kind of learning process performed by more than one person that takes place mainly in a virtual environment. According to this definition we should implement collaborative learning if and when there is a need for several people in a certain organization/ institute/university to learn together [3]. Moreover we should implement collaborative e-learning systems in order to satisfy the need of several people, in several places, in a certain organization/institute/university to learn together. The realistic visualization of the classrooms can only be accomplished by a 3D model of a virtual learning environment [4]. Furthermore to encourage more learners' participation, their better representation, more learner-to-learner interaction throughout community features, more contribution by the learners, less hierarchy and more empowerment, it is efficient to use collaborative virtual environments. Most of the commercial web-based training solutions lack sufficient realization of real-time communication features, meaning a shared sense of space and presence. In general, third party applications refer to features such as application sharing and video conferencing. Thus, there is a definite need for integrated solutions that can offer a

J. Fong et al. (Eds.): ICWL 2002, LNCS 2436, pp. 180-191, 2002.

much higher degree of usability and that do not demand additional technical requirements as in the case of most commercial systems. Therefore, we have to achieve the right balance among user requirements, learning methods and applied technologies and standards [5], with a view to implementing a system that is user centered and effective. Neither investigating new learning methods, nor inventing new technologies can accomplish the above requirement. We believe that a flexible and open system will satisfy the need of different user groups exploiting a bundle of technologies in a uniform and effective way. The application of virtual reality and the use of multi-user real-time communication platforms satisfy the need of efficient delivery of synchronous learning services. Collaborative virtual environments have drawn attention because they can provide learners and tutors with advanced interfacing capabilities and real-time communication support [6]. Motivated by these advantages, we have designed and implemented a prototype for collaborative e-learning using collaborative virtual environments. In this paper we focus on the implementation and design issues of the platform that support this prototype, as well as on the effectiveness of the prototype with respect to the collaborative e-learning application. We initially describe a virtual community for e-learning, its basic functionality and usage scenarios, which is based on collaborative virtual environments. We then describe the architecture and the main components of the platform that we have designed and implemented in order to support this virtual community. Afterwards, we discuss the effectiveness of our prototype in both the functional and technical point of view. Finally we present some concluding remarks and our vision for the next steps.

2 A Virtual Community for e-Learning

In order to implement a functional and effective e-learning virtual community, our first step is to investigate its main functional features. These functional features should differentiate an e-learning environment from other virtual environments (3D or not), which are designed and implemented for general use. According to [3] every virtual environment that has the following features can be characterized as an e-learning community:

- The environment should be explicitly designed. It can be visited by users, who have different roles and rights. It should be represented by various representation forms, which can range form simple text to 3D worlds. It should support various e-learning scenarios and have common features with a physical space
- The educational interactions in the environment should change the simple virtual space to communication space
- The learners in the environment should not be passive, but they should be able to interact
- The system that supports the e-learning environment should be able to integrate various technologies

According to the previous stated features of an e-learning virtual community the main requirements that should be met are the following:

- The e-learning environments should be based on templates that ensure a well-established community, able to handle users, e-learning material, user interaction, and different learning scenarios.
- The environment should offer various synchronous and asynchronous communication channels: chat, audio, e-mail, forums, shared objects, application sharing, gestures
- The environment should be populated by users who are represented by 3D avatars
- The environment should be aesthetic and easy to use

Actually the virtual community for e-learning is an integrated environment, which is based on a set of different virtual worlds that aim to offer to the participating entities the ability to navigate and interact in 3D shared space.

2.1 Usage Scenarios - Functional Specifications

The proposed system aims at offering innovative opportunities for the educational use of shared spaces for collaborative e-learning. The initialization of certain events when a user enters specific areas, the integration of sounds, the interaction through certain objects, the animation of objects in the virtual worlds and the interaction of the worlds with other applications enhances the user's sense of realism. The EVE provides the users with a shared sense of space, as all participants can be presented with the illusion of being located in the same place, such as in the same room, building, or terrain. It also supports a shared sense of time, meaning that participants are able to see each other's behavior as it occurs.

Based on the above, it is obvious that the EVE comprises a media over which educational procedures and interpersonal communication could be performed in a manner closer to the end users' need and perception. By simulating well-known everyday life procedures and actions, the EVE aims at presenting sophisticated services to the inexperienced user. Moreover, the environments could serve as a meeting point among the members of a learning community (e.g. university teachers and students). They provide the means for exchanging ideas, accessing amounts of information and collaborate on learning activities. Our EVE is designed in order to support two user scenarios. The first one is the collaborative e-learning scenario, which is more collaborative, open, unstructured and symmetric. The second one is the on-line lecture scenario, which is less collaborative, mainly tutor centered, more structured, more hierarchical

The main entities in the above scenarios are the learners, the tutor, the moderator, the shared objects and the educational material. According to the scenario the students and tutors have different access rights and authorities. Both the educational material and the shared objects can be manipulated by the previous three active entities (learners, tutor, moderator). The moderator participates only on the collaborative e-learning scenario. In the on-line lecture scenario the participating entities are the learners, the tutor, the shared objects and the educational material. The tutors' entities have the greatest access and amount of authorities and the students' entities have full access but limited authorities. The tutor would interact with the whole system through the appropriate user interface, which consists of a typical web browser, a VRML browser and a set of Java applets. This interface could give the tutor the ability to upload learning material to the available WWW server and to initiate and control learning sessions within the appropriate virtual worlds. On the other hand, learners will participate in the learning sessions through a different user interface with fewer capabilities. Both user interfaces are easy to use and need no specific knowledge from the point of the user.

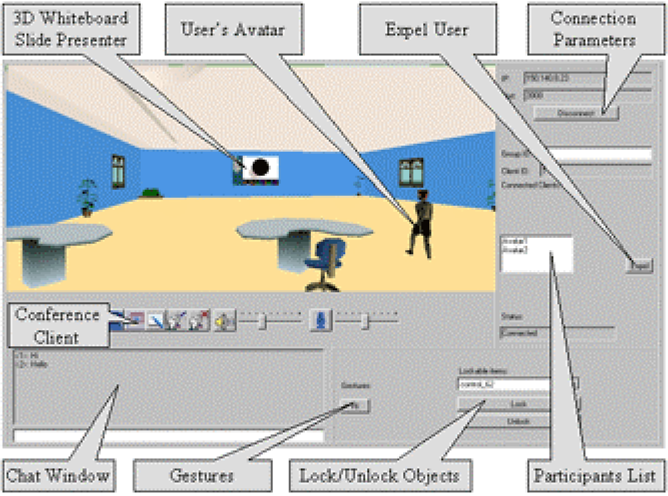

Fig. 1. User Interface

The functionality that the prototype offers to the learners/tutors is the following:
- 3D representation by articulated, human-like avatars
- Audio and chat communication as well as gestures to express feelings and actionsg
- Application/document sharing and collaboration on them
- 3D library with educational content and manipulation of 3D shared objects (if they are not locked by the tutor)
- Predefined animation, viewpoints
- 3D Slide presenter/whiteboard and 2D whiteboard

In addition, the tutors are able to
- Upload 3D/2D learning material in the 3D shared space
- Moderate the learning session by locking shared objects, and allowing/preventing the collaboration on the shared documents
- Expel an annoying student from the virtual world
- Lock/unlock shared objects to prevent/enable their manipulation by the learners

In the Fig. 1 the tutor's user interface is depicted.

In the collaborative e-learning scenario the participating entities are the learners, the moderator, the shared objects and the educational material. All the learners have the same access rights and authorities as in the on-line lecture scenario and in addition they can upload 3D/2D learning material in the 3D shared space, and locking shared objects. Aside from the learners capabilities the moderator can expel an annoying student form the virtual world.

3 Architecture and Components

In order to achieve the above goals our design is based on the following concepts: Scalability, consistency, extensibility and openness Also the proposed system should support several forms of data should be supported and embedded in the EVE. Furthermore, the users in an EVE should be able to communicate based on widely accepted conference standards (such as H.323 and T.120). An EVE should be a web-based application implemented with international accepted standards and technologies (HTTP, VRML[1], and JAVA). Finally the users should be represented by H-anim[2] compatible avatars, which support animation and gestures, for user representation.

The main step in the design phase was to specify the architecture of the system (Fig. 2), which should meet all the above-described functional and technical requirements in order to support the e-learning community in an effective way. The main concept of the proposed architecture was to divide the processing load of the necessary services of the e-learning community to a set of responsible servers. There are two types of divisions: (a) the division of the hosting of multi-user 3D worlds to set of communication servers (a set of these servers is called Message server), and (b) the division of the provision of specific functionality, into dedicated application servers.

The basic idea of our architecture is to divide the processing load of necessary services of an EVE (such as application sharing, chat and audio communication, educational content, etc.) to a set of servers [7] aside from communication of users or management of the virtual worlds as described in other models [8]. Furthermore, the structure of an educational community implies a the virtual environment can be separated into smaller parts, which are virtual rooms dedicated to a specific e-learning course. Therefore the lessons in the e-learning community are conducted in course 3D rooms. This provides a "segmentation" of the virtual community that enables us to design a communication model that consists of a number of message servers.

These types of servers are described in the following paragraphs.

3.1 Servers

As mentioned above the architecture is based on a set of servers, each designed to carry out a specific operation. These servers can be categorized in two main categories: Communication servers and application servers.

The communication servers (a set of these servers is called Message server) are responsible for the connection of the users and the consistency of the 3D shared space. Communication servers are:

- ConnectionServer, which handles the connection requests of the participants.
- InitServer, which holds the current state of the virtual community. When a new participant (client) arrives in the virtual world, it transmits the entire list of shared nodes that maintains, to the newly added client.

[1] Web 3D Consortium. The Virtual Reality Modeling Language (VRML) - Part 1, 1997. **http://www.web3d.org/technicalinfo/specifications/vrml97/index.htm** .

[2] Web 3D Consortium - Humanoid Animation Working Group H-Anim 1.1 specification. 1999, **http://h-anim.org/Specifications/H-Anim1.1/** .

- VrmlServer, which is responsible for sending update messages to the participants.

The application servers are responsible to offer specific functionality to the participants:

- ChatServer, which is responsible for the chat communication
- Conference Server, which is responsible for the application and data sharing, the whiteboard capability as well as the audio communication

Also the prototype uses an HTTP server, which contains all the necessary, HTML pages, VRML files, 3D object and other e-learning material.

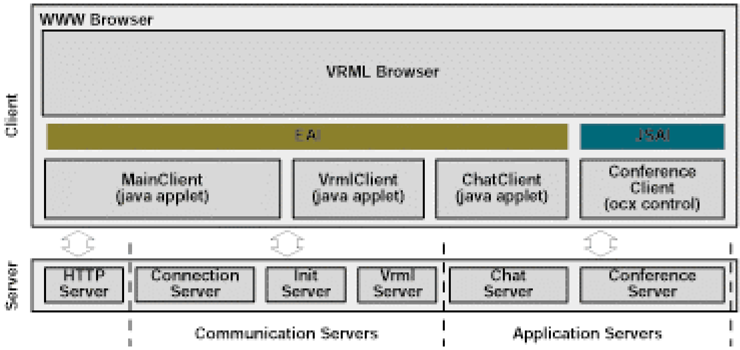

Fig. 2. Architecture and Components

3.2 Clients

The client, in order to communicate with the above-described set of servers, consists of six components: the web browser, the VRML browser, the MainClient, the VRMLClient, the ChatClient, and the ConferenceClient. The Vrml Browser is a plug-in, used to navigate user in the VRML world. Any VRML97, EAI[3] compliant browser could be used for this cause. Parallel Graphics' Cortona is a tested and recommended solution. The MainClient is a java applet that establishes and terminates the initial connection to Message Server. It presents the current connection status and a list of the participants populating the same 3D world. The VrmlClient is a java applet responsible for the interaction between the user and the 3D scene. It works in two phases. In the first phase, it receives all shared nodes from InitServer and initializes the VRML world. In the second phase, the VrmlClient sends and receives events to and from VrmlServer and updates the VRML scene according to the received events. In addition, this module gives the participants the capability to make predefined gestures and to lock/unlock shared objects. The ChatClient is a java applet that implements the exchange of chat messages among users in the same VRML world and it offers text conference functionality to the prototype. The ConferenceClient has a twofold role: it acts as an application sharing client and audio client. It allows

[3] Web 3D Consortium. External Authoring Interface (EAI), Final Draft International Standard, **http://www.web3d.org/technicalinfo/specifications/eai_fdis/index.html** .

participants to enter in an application sharing (T.120) and audio (H.323) conference that is established in the Conference Server. This module is an ActiveX control and is based on the Microsoft NetMeeting SDK[4]. It offers the participants the functionality of document/application sharing, 2D shared whiteboard, and collaboration on documents as well as audio communication.

3.3 Layer Model

According to [9] where a layer model for distributed virtual worlds is defined, the architecture of our prototype is based on layers shown in the following Fig. 3.

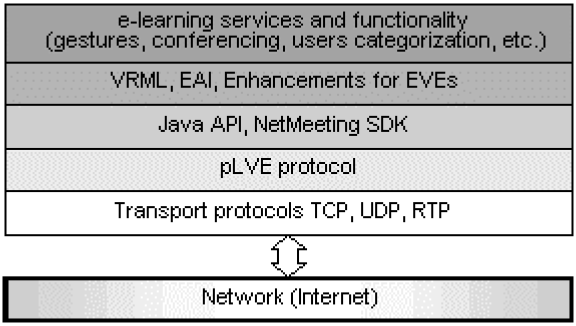

Fig. 3. Layer model

The underlying network is the Internet (without multicast capabilities) and UDP and TCP communication is used for the transmission of packets, according to the desired functionality. For example we use TCP communication for the initial connection to the ConnectionServer and UDP communication with the ChatServer. In addition we use the RTP protocol for the audio communication. On top of the network layer we have implemented a suitable application-specific protocol, which is called pLVE [10]. For sharing virtual worlds we use a java network interface and an enhanced VRML - EAI [7], suitable for maintaining a consistent 3D scene (distributing the necessary events) among all the participants over the network.

Finally, making the worlds shared we have implemented the rules for the interaction between users in order to provide the previous desired functionality such as gestures, users' roles and rights, lock/unlock objects etc.

4 Effectiveness of the Prototype with Respect to e-Learning

In the previous paragraphs we presented the features that an e-learning community must have, and we proposed an architecture that can support the needed learning scenarios. In the following paragraphs, we are going to perform a functional

[4] Microsoft Windows NetMeeting 3 Resource Kit, ©1996-2001 Microsoft Corporation.

evaluation of our Educational Community, which is based on the previously described architecture and we will discuss some technical issues with respect to the implementation of our prototype.

4.1 Functional Evaluation

The main aim during the design of our Educational Community was to achieve the required functionality in a simple and effective manner. We believe that we successfully tackled some important issues of an EVE, which we present in this paragraph.

Space Sharing, User Interaction and Avatar Representation. An EVE should be a place where different users can communicate with each other and with the environment as if they where in a real classroom. We accomplish the need to create the illusion that all users share the same space by transmitting all the events that occur, in the order they occur, to all of them. Furthermore, the existence of objects in the world that can be altered by the users, the shared objects, coupled with the feature of importing a new object into the world, give to our Educational Community a dynamic character, fulfilling the not-static nature of an EVE. Users can communicate with each other by sending either text messages or audio streams. Moreover, avatars' gestures provide a more realistic interaction among users, expressing when needed the emotion of each one to the others [11]. Regarding the avatars' representation in the educational community we focus on functions for representing oneself to others and for visualizing the others than for self re-presentation. Available functions in our prototype are:

- Perception: the ability of a participant to see if anyone is around
- Localization: the ability of a participant to see where the other person is
- Gestures: Representation and visualization of others' actions
- Identification: Each avatar has a unique number which is placed in his/her body in order for the other avatars to recognize him/her
- Social representation: Except the unique number in the avatar's body has placed a letter according to his/her status/task in the community. For this reason we use the letters T, L, and M for the tutors, learners, and moderators respectively.

User Categorization. In an EVE we need to have users with different roles and rights. For example a tutor must have more rights than a student and he must be able to perform a different role in the Educational Community. We accomplish this by providing each user with an access level that is set by the administrator during the creation of a new user's account. Furthermore, each shared object contains a variable that controls the access rights of that object. Thus, if a shared object has its access level equal to "STUDENT", then only users that have their access level equal or greater than "STUDENT" can access this object. That is not enough though. A user that is a tutor in a certain world might be a student in one or more other worlds. Since a different Message Server (that maintains its own list of users) serves each world, the achievement of this feature is not difficult by assigning a different access level to this user in each Message Server.

Learning Tools. Though both space sharing and user categorization is surely necessary in an EVE, they are not enough to characterize a Virtual Environment as a Learning Environment. As stated by its name, the most important feature an EVE should have is to contain tools suitable for teaching. In the prototype of our Educational Community we provide the following tools:

- A 3D Whiteboard: This whiteboard supports slide projection, line, circle and ellipsis drawing in a wide range of colors and text input in many sizes and colors. It also offers an UNDO last action and and CLEAR ALL previous action capabilities. The 3D Whiteboard is a part of the VRML scene and it can be seen through navigating in the world.
- A 2D Whiteboard: the 2D whiteboard is based on the Microsoft's NetMeeting whiteboard.
- A 3D Library: This library contains links to web pages that have educational material. A book on a shelf of the library represents each link. When a user clicks on a book, a web page is loaded in a separate window. The 3D Library supports dynamic addition and removal of books.

Locking/Unlocking Objects. There are cases where we need a certain object to be accessed only be one user at a time. For example when a user writes on the whiteboard nobody else must be able to write or delete something on it. We accomplish this by adding a variable named LOCK in each shared object. This variable controls the status of the object. The variable can take as values: -1 (the option of locking the shared node is disabled), 0 (the shared node is available for a user to lock it), >0 (The user with ID = LOCK has locked the shared node).

When a user, who locked one or more objects in a world, leaves the scene without releasing the lock, the objects are automatically unlocked. Moreover, a user with higher access level than the one that possesses the lock of an object can obtain the lock of that object from the other user.

Expelling Learner/Participant. Sometimes, a learner/participant can become annoying, preventing the smooth completion of a lecture. In these cases, the tutor (in the on-line lecture scenario) and the moderator of the class (in the collaborative e-learning scenario) has the ability to expel this user. The only action he/she has to take is to select the user from a list of all users participating in the class and to press the Expel button of the user interface. Clicking this button the user is disconnected of the system.

4.2 Technical Issues

In the process of developing our prototype, we encounter important problems such as the sharing of update messages, the initialization of the 3D worlds, the integration of the avatars, the audio communication and the scalability of the system. In this paragraph we will present some technical issues that came up during our effort to solve these problems.

Shared Events. As mentioned earlier, in order to achieve the illusion that all users share the same space, we have to transmit all the events that occur to all clients. At this point we had to deal with the following problem: What happens if an event-packet is lost during its transmission from the user’s client to the server and what happens if it is lost during its transmission from the server to a client. The first case is not dangerous, as the packet never reaches the server. Therefore, we have chosen to ignore this case and consider that the packet had never been sent. The user has the chance to re-send it if he wants to. The second case is the one that puzzled us the most. Since the event-packet reaches the server, it is transmitted back to all clients. Thus, if a client does not receive it, or receive it out of order, a problem of consistency among clients is immediately created, which puts in danger the stability of the system. We solved this problem in two parts. Firstly, every packet that reaches the server is numbered and then it is propagated to all clients. Every client that receives a new packet checks if its number is the expected one. If it is not, then the client requests the retransmission of the lost packet or packets. The second part of the solution covers the case where the server does not have any activity for a long time and as a result a client might have lost a packet and have not yet discovered it. In order to attain this matter, an empty numbered message, a NOP message, is sent every few seconds when there is no activity at the server.

Initialization of the Current State of the 3D World. Though transmitting Shared Events to all clients of the users populating a certain world is enough to maintain the consistency among them, the problem of initialization of a new client still remains. More specifically, the problem is the transformation of the 3D (VRML) scene of the newly added client in order to become identical with the scenes of all the others. One solution is to store all shared events in a file and then transmitting them to the new client would accomplish what we want. Unfortunately, this solution becomes less effective and inapplicable while the number of shared events increases as the time passes. In order to solve this problem we had to adapt a new concept; The SharedNode. A SharedNode is an abstract representation of a VRML node that contains only fields that we want to be able to be altered. These fields, which we call SharedEventIns or SharedEventOuts, are the representation of VRML EventIns or EventOuts respectively. Each SharedNode, SharedEventIn or SharedEventOut must have the same name as the VRML node, EventIn or EventOut that represents. This implies that only VRML nodes that were named using the DEF statement can become shared. Using SharedNodes, the procedure of initializing a new user becomes easy. We create a SharedNode for each VRML node we want to become shared. Then, we add the routes that have a SharedNode at least one of their ends and at the same time, we remove the corresponding route from the wrl file. In each instant, this set of SharedNodes, which we call SharedList, fully describes the present situation of the VRML scene because only fields of these nodes can be changed. Thus, the SharedList is the only information that a new user needs in order to update its VRML scene.

Avatars. The insertion of Avatars into the world created many problems for which the solution was not always so obvious. The fact that we wanted to have many duplicates of the same avatar, each having a different name in the same world, was the first problem we had to deal with. There was no way to assign a unique name to each Avatar, because the CreateVrmlFromUrl function did not support this feature. A different approach had to be followed. We created an empty node named with the DEF statement using the CreateVrmlFromString function. Then, we created the avatar using the CreateVrmlFromUrl function and we set it as a child to the named node. Thus, the requested feature was accomplished, as each avatar was wrapped in a node with a different name. A new problem appeared though. The access of a specific node of an avatar with the getNode() function was not possible because each duplication of an avatar had a node with the same name. This situation led us to create one more class; The SharedAvatar. A SharedAvatar inherits a SharedNode, thus, it has all the features of a SharedNode. Furthermore, it includes some special characteristics that help solve the previously described problem. Apart from the default SharedEventIns “Translation” and “Rotation” that determine the Avatars position in the world, it contains a pointer to a VRML node. This pointer is set during the initialization of a new SharedAvatar with the address of the Avatar’s node that has been created using the CreateVrmlFromUrl function. This pointer provides immediate access to the Avatar’s node, thus, we can use the “children” field of that node to access every node we need, avoiding the use of the getNode() function.

Audio Communication. The audio communication is based on client-server architecture. As described earlier the client is an ActiveX control, which is based on Microsoft NetMeeting. The server is a H.323 compatible conference server (actually we use MeetingPoint Conference Server). This server is a software Multipoint Control Unit (MCU) and allows three or more H.323 terminals to connect and participate in a multipoint conference. The MCU includes both multipoint controllers, which manage the H.323 terminal functions and capabilities in a multipoint conference, and multipoint processors, which process the audio, video, and data streams between H.323 terminals. For each room of the community we have established is a corresponding session in the conference server where the participants the specific room can automatically connect entering in the room.

Scalability. One of the main problems that we have encounter was the support large number of simultaneous users and/or provided services. This problem is very crucial for our prototype, which is designed for educational use, and it should be suitable for widespread use. For this reason the basic concept that affected the architecture of our prototype was the distribution of the workload. In order to divide the processing load of specific services we use different application servers such as the Conference Server, and Chat Server. In addition a separate Message Server (which consists of a ConnectionServer, an InitServer, and a VrmlServer) is used to serve each 3D world of the community. The separate sub-servers is used to carry out each of the specific operations of a Message Server. Following this approach, we achieve our initial goal of shattering into pieces the processing work and increasing the scalability of our system.

5 Conclusion - Future Work

In this paper we have presented a prototype, which targets the offering of e-learning services using collaborative virtual environments. This prototype is based on a research platform whose both design and implementation are based on the requirements of e-learning services. Implementing this prototype we have encounter not only technical problems, but also educational issues that help us use in a more efficient way the new ways of communication and interaction that distributed virtual reality technologies offer. These problems and their solutions have also been presented. Our next steps have to do not only with the integration of additional features and functionality to the system but also with improvement of scalability and networking characteristics. In more detail we would like to improve the scalability of the system supporting multicast communication between the MessageServers and offering multicast groups for each world of the community. Furthermore we would like to improve the stability of the system making each message server back-up server of the rest of message servers. From the functional point of view we would like to improve the avatars' representation including functions such as visualization of the interest focus, and communication with facial expressions as well as to develop a 3D brainstorming board in order to support collaborative learning techniques.

References

1. Bouras, C., Philopoulos, A., Tsiatsos, T.: e-Learning through Distributed Virtual Environments. In Journal of Network and Computer Applications, Academic Press, July 2001
2. Oliveira, C., Shen, X., Georganas, N.: Collaborative Virtual Environment for Industrial Training and e-Commerce. In Proceedings of Workshop on Application of Virtual Reality Technologies for Future Telecommunication Systems, IEEE Globecom 2000 Conference, Nov.-Dec.2000, San Francisco.
3. Dillenbourg, P.: What do you mean by collaborative learning? In P. Dillenbourg Collaborative-learning: Cognitive and Computational Approaches, Oxford: Elsevier, 1999. pp. 1-19.
4. Dickey, M.: 3D Virtual Worlds and Learning: An Analysis of the Impact of Design Affordances and Limitations in Active Worlds, blaxxun interactive, and OnLive!Traveler; and A Study of the Implementation of Active Worlds for Formal and Informal Education. Dissertation, The Ohio State University, 1999.
5. Bouras, C., Triantafillou, V., Tsiatsos, T.: Aspects of collaborative learning environment using distributed virtual environments. In Proceedings of the ED-MEDIA 2001 Conference, Tampere, Finland, June 25-30, 2001, pp. 175-178.
6. Bouras, C., Hornig, G., Triantafillou, V., Tsiatsos, T.: Architectures Supporting e-Learning Through Collaborative Virtual Environments: The Case of INVITE. In Proceedings of IEEE International Conference on Advanced Learning Technologies-ICALT 2001, Madison, Wisconsin, USA, August 6-8, 2001, pp. 13-16.
7. Bouras, C., Psaltoulis, D., Psaroudis, C., Tsiatsos T.: Protocols for Sharing Educational Virtual Environments. In Proceedings of 2001 International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2001) Split, Dubrovnik (Croatia) Ancona, Bari (Italy), October 09-12, 2001
8. Singhal, S., Zyda M.: Networked Virtual Environments: Design and Implementation. ISBN 0-201-32557-8, ACM Press, 1999.
9. Diehl, S.: Distributed Virtual Worlds, Foundations and Implementation Techniques Using VRML, Java, and CORBA. Springer-Verlag, Berlin Heidelberg, Germany, ISBN 3-540-67624-4, 2001.
10. Bouras C., Tsiatsos T.: pLVE: Suitable Network Protocol Supporting Multi-User Virtual Environments in Education. In Proceedings of International Conference on Information and Communication Technologies for Education, Vienna, Austria, December 6-9 2000, pp.73-81.
11. Capin, T., Pandzic, I., Magnenat-Thalmann, N., Thalmann, D.: Avatars in Networked Virtual Environments. John Wiley & Sons Ltd, West Sussex England, ISBN 0-471-98863-4, 1999.