uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Haifa Raja Maamar

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.A.Sc. (Electrical Engineering)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

A hybrid multicast transport protocol for collaborative virtual environments

TITRE DE LA THÈSE / TITLE OF THESIS

A. Boukerche

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Prof. E. Petrui

Prof. D. Petrui

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

A hybrid multicast transport protocol for collaborative virtual environments

By

HAIFA RAJA MAAMAR

A thesis submitted to the

Faculty of Graduate and Postdoctoral Studies

In partial fulfillment of the requirement for the degree of

Master of Applied Science in Electrical and Computer Engineering.

The Ottawa-Carleton Institute for Electrical and Computer Engineering

School of Information Technology and Engineering

University of Ottawa, Ontario, CANADA

Your file Votre référence
ISBN: 978-0-494-50904-3
Our file Notre référence
ISBN: 978-0-494-50904-3

Abstract

In recent years, we have witnessed a growing interest in the synchronous collaboration based class of applications. Several techniques for Collaborative Virtual Environments (CVE) and Collaborative Haptic, Audio and Visual Environments (C-HAVE) have been designed. However, several challenging issues remain to be resolved before CVE and C-HAVE technologies become a common place.

In this thesis, we focus on applications that are based on closely coupled and highly synchronized haptic tasks that require a high-level of coordination among the participants. Four main protocols have been designed, in the literature, to resolve the synchronization issues in collaborative virtual environments: the Synchronous Collaboration Transport Protocol (SCTP), the Selective Reliable Transmission Protocol (SRTP), the Reliable Multicast Transport Protocol (RMTP) and the Scalable Reliable Multicast (SRM). However, none of these protocols has been able to meet all of the basic CVE requirements, i.e., scalability, reliability, synchronization, and minimum delay.

In this thesis, we propose a hybrid multicast transport protocol, which we refer to as HMTP that aims at meeting the CVE requirements. We describe our protocol, discuss its implementation and report on the performance results we have obtained for two virtual tele-surgery training class of applications.

The following publications by the author are relevant to this thesis.

Journal:

1) A. Boukerche, H. Maamar, A. Hossain, "*An efficient hybrid multicast transport protocol for collaborative virtual environment with networked haptics*", ACM/Springer Multimedia Systems, Vol. 13, pp. 283-296, 2007.

Conference:

1) A. Boukerche, H. Maamar, A. Hossain, "*A performance Evaluation of a hybrid multicast transport protocol for a distributed collaborative virtual simulation of a brain tumor tele-surgery class of applications*", The 12th IEEE Symposium on Computers and Communications (ISCC'07), pp. 975-980, 2007

2) A. Boukerche, H. Maamar, A. Hossain, "*A design and analysis of a Hybrid Multicast transport protocol for the haptic virtual reality tracheotomy tele-surgery application*", IEEE Proceedings of the 21st IEEE Int'l Parallel and Distributed Processing Symposium, Long beach, CA, pp. 1- 6, 2007

3) A. Boukerche, H. Maamar, "*An efficient Hybrid Multicast Transport Protocol for collaborative virtual environment with networked haptics*", IEEE International Workshop on Haptic Audio Visual Environments (HAVE), pp. 78-83, 2006

TO MY PARENTS...

Acknowledgment

I am indebted to my supervisor Dr. Azzedine Boukerche, my family, and my friends whom I could not have completed my thesis without their continuous support.

First, I would like to express my sincere gratitude to my supervisor Dr. Azzedine Boukerche, who specifically helped me over the past year in the knowledge that I was researching and writing this thesis. I could not have had a better supervisor and could not have asked of more time as he was always listening to my ideas, motivating me to work harder, and challenging me when I needed to be challenged. He believed in me and guided me towards my academic success. Thanks for his continuous help and his financial support during my graduate studies at PARADISE Research Laboratory. This work was partially supported by ORNEC and NSERC research funds. Thanks are also in order to Prof. N. Georganas and Mr. Francois Malric for allowing me to use DISCOVER Research Laboratory facility including the haptic devices that I used in my experiments.

My family deserves greater thanks. Without their continuous love, sacrifice and support, I would not be where I am today. I am thankful to my Mom and Dad, brothers Walid and Sofien for their constant encouragement.

Finally, I warmly thank Abed Matar and Oualid Abidi for their constant support throughout my graduate studies. I am so grateful for their suggestions, encouragement and the time they devoted to my thesis.

Haifa Raja Maamar
University of Ottawa
February 2008.

Contents

LIST OF FIGURES

LIST OF TABLES

List of Acronyms

ACK: Acknowledgment

C-HAVE: Collaborative Haptic Audio Visual Environment

CVE: Collaborative Virtual Environment

DIS: Distributed Interactive Simulation

DLL: Dynamic Loadable Library

DOF: Degree Of Freedom

DR: Designated Receiver

DVS: Distributed Virtual Simulation

GHOST: General Haptics Open Software Toolkit

GUI: Graphical User Interface

HLA: High Level Architecture

HMD: Head Mounted Display

HMTP: Hybrid Multicast Transport Protocol

IETF: Internet Engineering Task Force

IRTF: Internet Research Task Force

JNI: Java Native Interface

LAN: Local Area Network

MFTP: Multicast File Transfer Protocol

MGM: Multicast Group Management

LES: Loss Estimation System

MMOSG: Massively Multiplayer Online Social Games

MPEG: Moving Picture Experts Group

MTP: Multicast Transport Protocol

MTP-2: Multicast Transport Protocol-2

NACK: Negative Acknowledgment

PMATS: Predator Aircrew Mission Training System

QoS: Quality of Service

RAMP: Reliable Adaptive Multicast Protocol

RBP: Reliable Broadcast Protocol

RMP: Reliable Multicast Protocol

RMTP: Reliable Multicast Transport Protocol

RTCP: Real Time Control Protocol

RTI: Run Time Infrastructure

RTP/I: Real Time Application Level Protocol for Distributed Interactive Media

RTP: Real-Time Transport Protocol

SCTP: Synchronous Collaboration Transport Protocol

SM: Session Manager

SRM: Scalable Reliable multicast

SRTP: Selective Reliable Transmission Protocol

S-SCTP: Smoothed Synchronous Collaboration Transport Protocol

TCP: Transmission Control Protocol

TMTP: Tree-based Multicast Protocol

TRAM: Tree-Based Reliable Multicast Protocol

TTL: Time to Live

UAV: Unmanned Aerial Vehicles

UDP: User Datagram Protocol

VE: Virtual Environment

VR: Virtual Reality

VRML: Virtual Reality Modeling Language

WAN: Wide Area Network

XTP: Xpress Transport Protocol

Chapter 1

Introduction

Nowadays, Collaborative Virtual Environments (CVE) are used in many scenarios such as tele-surgery [36], military training [54], entertainment [38], industrial training [37], just to mention a few. All of these applications require that the users collaborate in closely coupled and highly synchronized tasks to manipulate shared objects. These tasks require a high level of coordination among them that is applied by guarantying that update messages corresponding to the change of the state are communicated reliably and in a timely manner between users. By adding the notion of "haptic" to CVE, the degree of synchronization has become higher and the design of a protocol that satisfies the CVE requirement and ensures efficient communication has become urgent. Therefore, an efficient communication approach is required for haptic CVE class of applications.

1.1. Problem Statement

One of the main issues that CVE applications encounter is the lack of synchronization due to packet delay, loss, and jitter [17]. In fact, due to network limitations and traffic conditions, some of the packets sent are either lost or delayed. This, unfortunately, may lead to a lack of synchronization among the participating users. To overcome this problem, several strategies have been proposed [2, 3, 8, 9, 17].

However, all of these proposed protocols have failed to resolve the basic synchronization issue and satisfy all of the CVE requirements, i.e., reliability, minimum delay, scalability, and synchronization. To the best of our knowledge, each of these protocols either concentrates on one aspect and/or fails to meet the other requirement that are necessary for CVE applications. Moreover, most of the protocols designed have not satisfied the end-to-end delay as defined by the collaboration environments [8,9], which cannot exceed 200 msec. Among the protocols designed in this area, the following protocols have been used in CVE: SRTP [1, 2], SRM [10], RMTP [7], and SCTP [8, 9].

1.2. Motivation and Contributions

Synchronous collaboration has been the subject of much research in recent years and will continue to be, as long as there is a need to have protocols that can ensure the CVE requirement: reliability, minimum delay, scalability and synchronization.

In this thesis, we propose a hybrid multicast transport protocol that combines four existing transport protocols while taking advantage of the features of each of the chosen protocols. The four protocols used for our design are the following: SRTP [1, 2], SRM [10], SCTP [8, 9], and RMTP [7].

Our protocol is based on the *client-server* architecture, and uses a *multicast tree* in order to avoid the congestion and delay problems when sending messages. In terms of transmission, our proposed protocol uses three modes of transmission to ensure the reliable transport of the data. Our main contributions are to combine the protocols SRTP [1,2], SRM [10], RMTP [7], and SCTP [8,9] and to add a timer that allows the receivers to detect a loss or a delayed packet. Throughout this thesis, we shall refer to our scheme the Hybrid Multicast Transport Protocol or HMTP. We developed, implemented

and tested our protocol in two tele-surgery applications, i.e., brain tumor and tracheotomy tele-surgery applications. In this thesis, we discuss the set of experiments we have carried out to evaluate the performance of our scheme, and report on the results we have obtained.

1.3. Thesis Outline

The remainder of the thesis is structured as follow:

- ✓ Chapter 2 introduces the Collaborative Virtual Environments by presenting its requirement, architectures, standards, and application, followed by an overview of the protocols designed for the Collaborative Virtual Environments' class of applications.
- ✓ Chapter 3 introduces the multicasting problematic in general followed by an overview of the multicast transport protocols proposed in the literature.
- ✓ Chapter 4 discusses the design and implementation of our proposed protocol: Hybrid Multicast Transport Protocol (HMTP)
- ✓ Chapter 5 discusses the implementation of the HMTP for the two tele-surgery applications that were carried out jointly at the DISCOVER and PARADISE research laboratories.
- ✓ Chapter 6 exhibits the results obtained when comparing HMTP with S-SCTP and illustrates the performance of HMTP in terms of reductions in both execution time and the occurrence of failures.
- ✓ Chapter 7 concludes the thesis and discusses potential and future directions of this work.

Chapter 2

Collaborative Virtual Environments Using Haptic Devices

2.1. Introduction

The aim of this chapter is to provide an overview of the protocols designed for the Collaborative Virtual Environments class of applications. This chapter is structured as follows. In the first two sections, we introduce the Virtual Reality and Haptic systems. Next, the Collaborative Virtual Environments requirement, architectures, standards and applications are presented in the third section. In the last section, we describe several techniques designed for Collaborative Virtual Environments.

2.2. Virtual Reality

Virtual Reality (VR) is a technology that allows a user to interact with a computer-simulated environment using specialized peripherals. This technology was specially the subject of interest of media and science fiction novels such as *Neuromancer* [35] by Gibson. Virtual reality (VR) has been the topic of intense research literature in the last

decade. In the Encyclopedia of Virtual Environments [25], the Virtual Reality (VR) is defined as follow:

> *"Virtual Reality is a human-computer interface in which the computer creates a sensory-immersing environment that interactively responds to and is controlled by the behavior of the user."*

In his book Neuromancer [35], Gibson defined the Virtual Reality as:

> *"A consensual hallucination experienced daily by billions of legitimate operators, in every nation... A graphic representation of data abstracted from the banks of every computer in the human system. Unthinkable complexity. Lines of light ranged in the non-space of the mind, clusters and constellations of data..."*

Virtual reality (VR) has many synonyms in the literature, such as Virtual Environment (VE), Virtual Presence, Artificial Reality, Cyberspace, and Virtual Worlds [58]. Most current Virtual Environments (VE) are visual experiences, displayed most of the time on a computer screen, but some simulations include additional sensory information, such as sound through speakers or headphones. Users can interact with the VE either through the use of input peripherals such as the keyboard and mouse or through advanced devices such as haptic devices. These haptic systems improve the user's interaction with the VE and give him a sense of "touch" and force feedback when manipulating virtual objects. Many haptic devices are being used nowadays in VE and are offering more sensitive and more realistic way to exchange information with the VE.

2.3. Haptic Systems

During the last decade, the science of haptic has gained an important attention, and the applications using this technique have been wide spread within the VR community. The word "haptic" was defined to be [30, 32]:

> *"The capability to sense a natural or synthetic mechanical environment through touch."*
>
> *"The study of the sense of touch in human-computer interaction."*

The haptic device enables the user to manipulate virtual objects in a natural way and to feel the stiffness and texture of these objects [32].

Many definitions of the haptic technology were given in the literature and the most common one is given by Wikipedia [78] below:

> *"Haptic technology refers to technology which interfaces the user via the sense of touch by applying forces, vibrations and/or motions to the user."*

Virtual environments are the interaction of a user with a computer simulation. This interaction can be either through input devices such as the mouse or keyboard, or through haptic devices, which provide a sense of "touch" and "realism" and a force feedback to the user. This feedback is accomplished by applying a degree of opposing force to the user along the x, y and z axis.

Haptic technology is gaining a growing interest in the VE applications and the use of haptic devices is being a key part of these applications. Such devices are divided into two-dimensional and three-dimensional devices. Figure 2.1 illustrates a classification of the haptic devices.

2.3.1 The Two-Dimensional Haptic Device

These devices are used to assist users who are visually disabled. They offer a resistance at the edges of windows and buttons in order to enable the users to sense the Graphical User Interface. Such devices include:

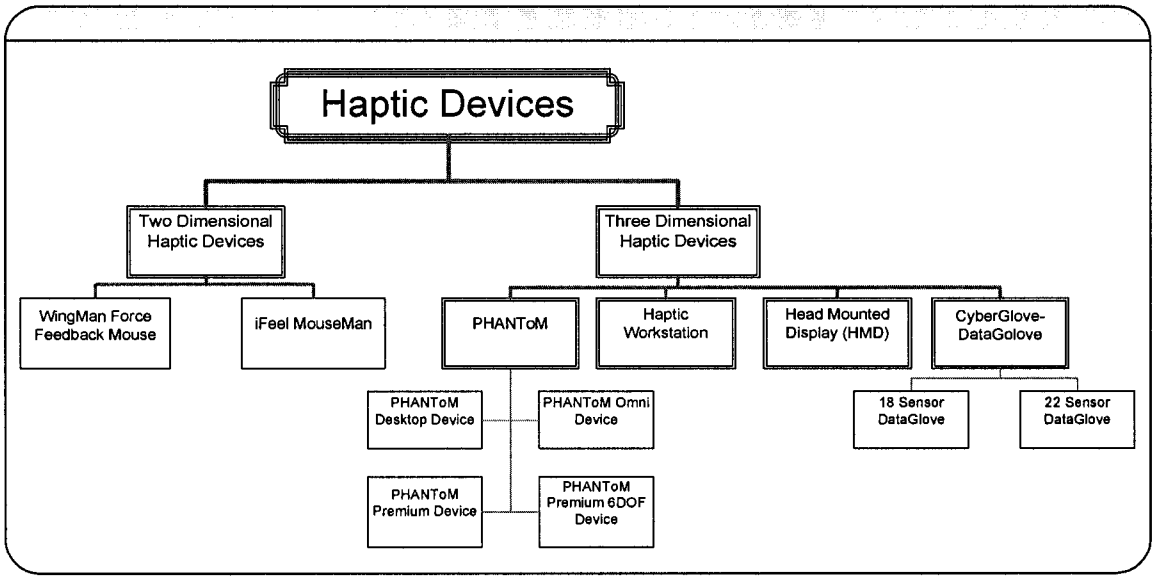


FIGURE 2.1: HAPTIC DEVICES

a) The *WingMan Force Feedback Mouse* was designed by Logitech and allowed users drawing a 2D image to sense the cursor and interact with points and lines. This device never gained widespread acceptance. In fact, in order to simulate the force, the mouse was always attached to a plastic base and required a separate power supply. Moreover, this device was very expensive and unnatural for daily use [26].

b) The *ifeel MouseMan* was designed by Logitech International S.A [60] in order to fix the problems that the WingMan Force Feedback Mouse [26] encountered. It has a sensory motor [27] inside which gives a tactile

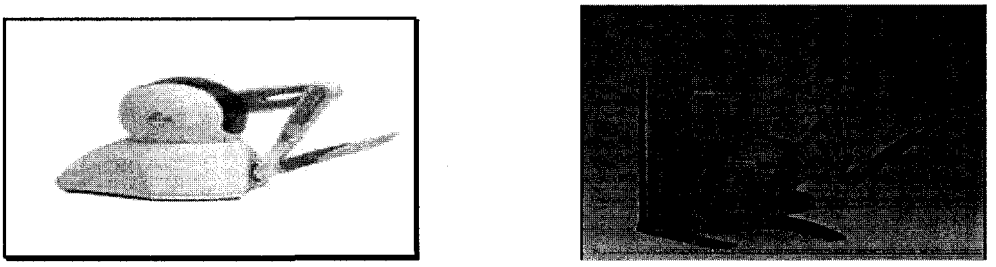
feedback by vibrating on icons, buttons, and hyperlinks. This mouse is entirely powered by a USB cable and does not need a separate power supply [26].

2.3.2. The Three-Dimensional Haptic Device

These devices are used for applications having pedagogical perspectives and aiming to train users on different scenarios depending on the application. These scenarios include tele-surgery training, military training, industrial training [8, 9, 17, 20] just to mention a few. The Three-Dimensional Haptic Devices permit to the user to "touch" and manipulate 3D virtual objects.

a) The *PHANToM* is designed by *SensAble* Technologies, Inc. [28]. Different models of this haptic device exist and this to meet needs of the different applications. These models include:

(i) The *PHANToM Omni Device* [61], illustrated in Figure 2.2 (A), is the most cost effective haptic device available [61]. It gives 6 degree-of-freedom (DOF) capabilities and ensures quick installation and a simple use.



(A) *(B)*

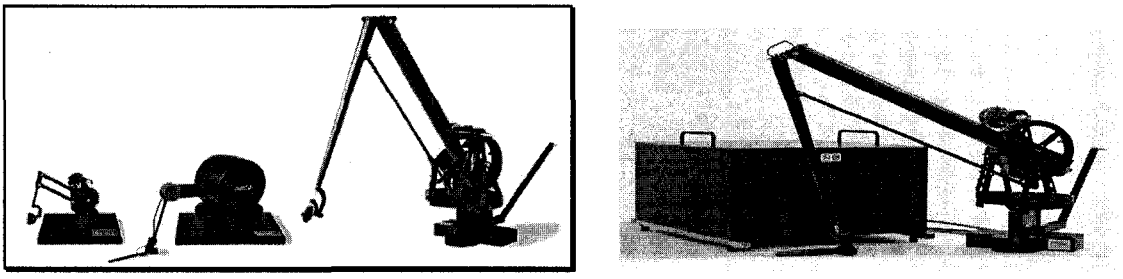
FIGURE 2.2: PHANTOM OMNI DEVICE (A), PHANTOM DESKTOP DEVICE (B)

(ii) The *PHANToM Desktop Device* [63] is illustrated in Figure 2.2 (B). This haptic device offers the ability to have a précised positioned input thanks to its 6 degree-of-freedom positional sensing. It gives an easy and sure way to tie different haptic devices to the standard PHANToM Desktop stylus.

(iii) The *PHANToM Premium Devices* [62] is composed of three different models illustrated in Figure 2.3(A):

- ✓ Premium Models 1.0, 1.5, 1.5/6DOF, and 3.0 [62]
- ✓ Premium Models 1.5 High Force and 1.5 High Force /6DOF [62]
- ✓ Premium Model 3.0/ 6DOF [62]

These devices offer 3 DOF positional sensing and 3DOF force feedback. The PHANToM Premium 1.5/6DOF, 1.5 High Force /6DOF, and 3.0/6DOF [62] can be used in applications aiming teleoperation [62], molecular modeling [62, 79], and virtual prototyping [79].



(A) *(B)*

FIGURE 2.3: PHANTOM PREMIUM DEVICES (A), PHANTOM PREMIUM 6DOF DEVICE (B)

(iv) The *PHANToM Premium 6DOF Devices* [64] are composed of three different models

- ✓ The PHANToM Premium 1.5/6DOF [64]

- ✓ The PHANToM Premium 1.5 High Force/6DOF [64]
- ✓ The PHANToM Premium 3.0/6DOF [64]

These devices offer 3 translational DOF force feedback. Moreover, they give torque feedback in 3 rotational DOF in the three axes. The PHANToM 3.0/6DOF [64] has the ability to move as a full arm movement rotating at the shoulder. The latter device is illustrated in Figure 2.3(A).

b) The *CyberGlove-Data Glove* [29] is an instrumented glove that offers up to 22 high-accuracy joint angle measurements.

It uses a technology that converts fingers and hand motions to a digital joint-angle data. This haptic device includes 2 models:

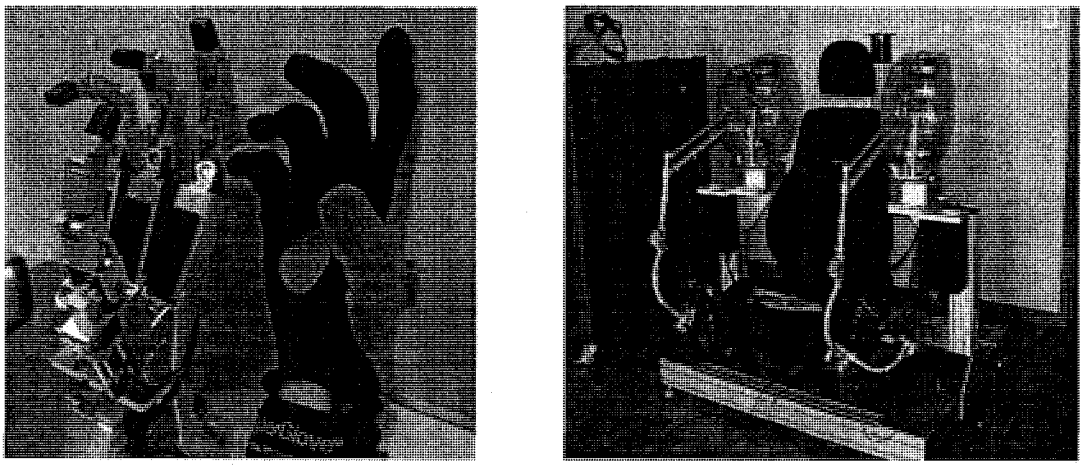
(i) The 18-sensor model [29] has two bend sensors on each finger, four abduction sensors, sensors measuring thumb crossover, palm arch, wrist flexion and wrist abduction.

(ii) The 22-sensor model [29] has three flexion sensors per finger, four abduction sensors, a palm-arch sensor, and sensors to measure flexion and abduction.

In order to measure the position and the orientation of the arm in the space, 6DOF tracking sensors [29] are included in the glove. When the arm moves, the glove detects the movement and sends an electrical signal to the computer, which transforms it into the virtual environment [25].

The CyberGlove [29], illustrated in Figure 2.4 (A), has been used in many applications including virtual reality biomechanics [65], and animations [65]. An enhanced version of the CyberGlove is the *CyberGrasp* which adds a resistive force feedback to each finger and which is used especially in virtual reality simulations. The CyberGrasp is a Glove-style haptic interface and an innovative force feedback system for the fingers and hand. This device allows the user to "touch" and manipulate virtual objects and to feel realistic force feedbacks.

c) The *Haptic Workstation* [31], illustrated in Figure 2.4 (B), involves the use of the following feedbacks: the *sight, sound* and *touch.*



(A) *(B)*

FIGURE 2.4: CYBERGLOVE-DATA GLOVE (A), HAPTIC WORKSTATION (B)

The Haptic Workstation system [31] is a combination of:

(i) A right-hand and a left-hand force-feedback

(ii) A structure with vertical adjustability for sitting and standing applications.

(iii) A Head-mounted display combined with a head tracker used for 3D viewing

(iv) Adjustable automobile seat

The *Haptic Workstation* [31] is specially used for medical training [78], and body mechanics and rehabilitation [31].

d) The *Head-Mounted Display (HMD)* [56] is a visor or eye-glasses that embed one or two small displays with lenses. This display device can present a different image to each eye and has the ability to improve the realism of the images and this is due to its distant focus feature [56]. Some HMDs offer the see-through imagery feature and were specially used for gaming applications. Other applications, such as the military and firefighters [57] use HMDs that offer to display tactical information, i.e., thermal imaging data. Figure 2.5 illustrates the Head Mounted Display.

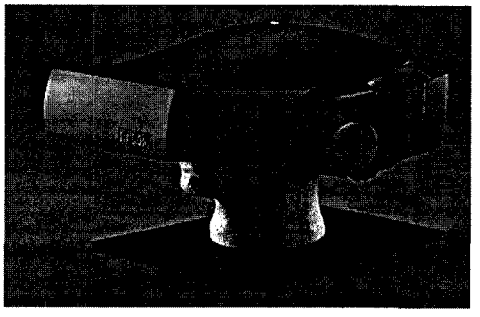


FIGURE 2.5: HEAD-MOUNTED DISPLAY

As described above, several haptic devices were designed in order to satisfy the different applications targets. Some of these devices have training purposes, such as military [57],

firefighters or medical training applications [78]; some others were used for tele-operation [62], molecular modeling [79], virtual prototyping [79] or even animations. In order to be able to meet the different requirement, VE had to improve and to add some notions. These notions include the "Collaboration". The latter was added to VE in order to meet the requirement for applications that necessitate users to collaborate together and to manipulate shared objects. The following section 3 will explain the notion of Collaborative Virtual Environment, its requirement and Standards, and a classification of its applications.

2.4. Collaborative Virtual Environments

The notion of "Collaboration" has been added to the VE in order to allow remote users to share a virtual reality environment and to participate in a simulation. The Collaborative virtual environments (CVEs) require that the users collaborate in closely and coupled highly synchronized tasks to manipulate shared objects. Many definitions for CVEs were given in the literature; here is one of the definitions given by Gibson in his book Neuromancer [23].

> "*Collaborative Virtual Environments (CVEs) are distributed virtual reality systems that offer graphically realized, potentially infinite, digital landscapes. Within these landscapes, individuals can share information through interaction with each other and through individual and collaborative interaction with data representation.* "

However, the most complete CVE definition was given by Elizabeth F. Churchill in her book [23] and she defines it as follow:

"Collaborative Virtual Environments (CVEs) are computer-enabled, distributed virtual spaces or places in which people can meet and interact with others and with virtual objects".

By adding the notion of "Haptic" to CVE, the manipulation of the virtual objects has improved and this is due to the sense of "touch" that the haptic devices offer. These new environments, that support the collaborative touch in the VE, are called the *Collaborative Haptic Audio Visual Environments (C-HAVE)* [32]. X. Shen *et al.* [32] have defined C-HAVE as follows:

"C-HAVE is a network of nodes, where each node corresponds to a computer whose operator is part of the shared virtual environment and that is responsible of some virtual objects."

Users in the C-HAVE are distributed geographically. They are either active collaborating by the use of different kinds of haptic devices, or they are just passive observers [32]. If only one user manipulates a virtual object, then it is called *unilateral tele-haptics*. However, when multiple users haptically interact with one same object, then it is called *bilateral tele-haptics* [32].

Several CVEs and C-HAVE applications have been designed [35, 36, 37, 38 and 57]. These different applications require having the following components:

- ✓ Shared virtual environment
- ✓ A graphic rendering program
- ✓ An interface with haptic devices

The following section will give a detailed description of the different components of a CVE or C-HAVE application.

2.4.1. CVEs Components

CVE applications allow multiple remote and distributed users to collaborate together and manipulate shared objects. In order to work together efficiently, users need to share the virtual environment. To that end, many CVEs replicate the virtual environment on each user's machine and use a control consistency method to synchronize the actions performed by the different participants. To achieve collaboration, a CVE application has to have the following components:

1) Shared Virtual Environment: The virtual environment is the description of the 3D world and objects specifying the shape of the objects, their colors, textures, lighting, and positions. The VE can be designed using several software such as 3DS MAX [66], OpenGL [67], Direct3D [68], and the description of the virtual objects can be obtained by using either VRML (described in the Section 2.4.3) or JAVA 3D which is a scene graph based-3D API that runs over OpenGL and Direct3D.

The 3D objects can be either active or just passive and have no behavior. An object is active if a user processes it and changes its states and behavior.

Sharing the virtual environment consists of sharing virtual objects and interacting with them in real time. As mentioned earlier, in order to achieve collaboration and to manipulate shared objects, the virtual environment has to be shared by the different users participating in the application. To that end, the virtual environment, as well as the shared objects and avatars, are replicated on the machine of each user. However, it is important to keep consistent replicas and to maintain the synchronization between the participants. To that end, a good transport protocol and a control consistency method are used.

2) A Consistency Method: As mentioned above, a good consistency method is needed in order to maintain the synchronisation, concurrency, and causality of the actions or events performed by the participants. The synchronisation deals with the coordination of the events. The concurrency of the events aims to have events occurring simultaneously; while the causality ensures that the events are ordered using a causal order. Behaviours and events have to be synchronised constantly during the simulation. To that end, a system clock on each user's machine is used. A good consistency method improves the collaboration and interaction among users.

3) A Computer Graphic Rendering Program: Rendering is the process of generating an image from a model that is a description of a 3D object. This process allows obtaining a final image by adding the texture, colors, lights and position to the 3D object. Many techniques have been designed in order to facilitate the rendering process and to obtain a good final image. Animations and behaviors can be added to the objects, and when they are rendered, the user sees objects moving.

4) The Haptic Device Interface: A haptic device interface allows the interaction between the application users and virtual objects. The Personal Haptic Interface Mechanism PHANToM [61] gives a force-reflecting interface between the user and the computer. SensAble Technology Inc. [28] offers a General Haptics Open Software Toolkit (GHOST ® SDK) [70], which is a C++ software toolkit that helps developing touch-enabled applications. GHOST SDK [70] provides libraries of 3D prismatic and polygonal objects, and force effects that can be used to add persuasive physical dimension to the

applications. SensAble Technology Inc [28], also provides OpenHaptics toolkit API [71] that allows to add 3D navigation to the applications. OpenHaptics toolkit API [71] is designed to incorporate libraries such as physics and collision detection engines.

A CVE application needs to have the above mentioned component in order to ensure the collaboration. However, these components do not ensure a successful execution. The application still needs to satisfy CVEs requirement in order to provide collaboration among users. The next section will give a detailed description of the CVEs requirement.

2.4.2. CVEs Requirement

Several applications and techniques have been designed using the CVEs. However, in order to have a successful execution, these applications have to satisfy the following requirement:

1) Reliability: In Collaborative Virtual Environments, the reliability of data transfer is important. In fact, the state of graphical objects, the state of participants, the events performed are crucial and have to be transferred most of the time in a reliable manner. It is important that all the participants have the same view of the simulation running.

Several protocols were designed in order to achieve the reliable transfer of data. These protocols ensure that, when a message is transferred, no packets are lost and the receivers get the entire message. Among these protocols we mention: TCP and RMP.

Several systems use TCP in order to ensure the reliability. This protocol guarantees the reliable and ordered delivery of all the packets sent. However, this is not always necessary. In fact, transient data that change frequently, such as position, do not need to be delivered reliably. Only the final position has to be sent reliably. TCP does not support this feature, and the use of this protocol is then not a good solution most of the time.

2) Minimum Delay: The delay is an important issue in distributed collaborative virtual environments applications. It is defined as the roundtrip delay of a packet in the network. Delay outcomes from the network used to transmit messages, and the processing of these messages at the endpoints [34]. The network delay is a combination of:

i. The transmission delay: is the amount of time required to transmit the packet from the sender to the receiver.

ii. The switching delay

iii. The queuing delay: is the sum of delays encountered by a packet between the time of insertion in the network and the time of delivery to the addresses.

iv. The retransmission delay: is the amount of time required to retransmit the packet to the receiver.

There are two aspects of the delay to be considered:

i. The latency: is the time delay between when an event is initiated and when it is received by another user.

ii. The jitter: is the variation in the latency.

Delays can have intense consequences on collaboration, especially on the feedback, the communication, and the coordination. In fact, the feedback can be delayed for some users: participants may think that their actions have failed and will repeat them again. Moreover, users will have different outputs, due to the fact that the events happened in a different order for some of them. For these reasons, systems that use CVEs, have to maintain a minimum delay in order to have a successful execution.

*3) **Scalability:*** The scalability is an important property that the distributed and CVEs environments applications have to satisfy. An application is scalable, if it is able to handle growing amount of users in a graceful manner. It has to be efficient in expanding its resources to accommodate a big number of participants. The main reason to design a scalable application is reduced cost and effort. In fact, if an application has to be redesigned every time a new user joins, the application will be in a continuous redesign. Several protocols were designed in order to ensure the scalability. These protocols include: SRM (Scalable Reliable Multicast).

*4) **Synchronization:*** Synchronization is the most important requirement for the success of an application using CVEs. The synchronization is a problem of timekeeping which deals with the coordination of events in an application. For the collaboration, it is crucial to have an application that satisfies the synchronization. In fact, if for example, we are running a tele-surgery application: two surgeons are performing a surgery on a patient. It is important to have coordinated events in both sides. Otherwise, if the events are not synchronized, one surgeon will think that the other did not perform a task, and will do it from his side, which results on the task being performed twice, and the patient hurt. If the events are not synchronized in both sides of the users, then the output is not the same for both of them.

2.4.3. CVEs Standards

Systems and applications based on CVEs have known an important increase and the need to unify standards such as data exchange or network exchange between applications has become urgent.

*1) **RTP/I:*** Real Time Application Level Protocol for Distributed Interactive Media (RTP/I) [39] is a standardized application level protocol framework that targets to let applications to be interoperable and to allows the reuse of key functionality in form of generic services [35, 39].

RTP/I has been designed in order to satisfy the following requirement [39]:

i. Framing of event and state data: is ensured via the reliability of the data transfer, the ordering, and the timing.

ii. Support for consistency and fragmentation: all of the participants keep a consistent and synchronized state of the medium.

iii. Flexibility: is ensured via the ability to use different reliability mechanisms

iv. A standardized way to query the state of a sub-component: an application needs sometimes to have the state of some sub-components. The RTP/I has a standardized query to have this information.

v. The ability to convey meta information: important information such as quality reports, or information about the participants, the medium or sub-components are transmitted on a standardized way by an application level protocol.

*2) **The DIS Approach:*** Distributed Interactive Simulation (DIS) [74] is a group of protocols designed by the US Department of Defense [35]. DIS is a standard for interconnecting large numbers of simulators across local and wide area networks. It is also a standard protocol for simulator interoperability. It combines the following features within a single shared synthetic environment [35]:

i. *Object/event architecture*: standardized network packets called *PDU* are transmitted and that contain the status and actions of the entities.

ii. *Autonomous simulation nodes*: simulation nodes transmit the objects events of whom they are responsible.

iii. *Transmission of state information only*: state modifications are transmitted on a reduced update rate. It is for this reason that DIS is based on predictive techniques, such as the dead-reckoning technique, that allow users to predict the future state of an object based on his past states.

Although DIS is a standard protocol for simulator interoperability, it has three major drawbacks [35]:

i. Due to the use of UDP/IP, messages can arrive in a wrong order.

ii. Due to the broadcast, the scalability is not easy to establish.

iii. The messages sent are part of standardized PDUs of fixed size.

3) The High Level Architecture (HLA): HLA [75] is a standard that allows the interoperability of heterogeneous simulations and that reprocesses simulations and their components [35, 75]. This standard focuses on four concepts:

i. The Run Time Infrastructure (RTI): is a distributed operating system that is responsible of the communication between the all simulation models.

ii. The Interface specification: is a formal description of the interface that exists between the HLA application and the RTI.

iii. Technical rules: are a set of rules that the HLA user has to follow.

iv. Object Model Templates: are standardized formats that allow specifying the simulation models functionality and the interaction between models.

HLA is composed of federations that concentrate on specific areas of the simulation [35]. These federations are populated by federates, which are members that can be simulation models, simulators, passive viewers, autonomous agents, or data collectors. Simulated entities are defined as objects. In order to minimize the network traffic, HLA offers two mechanisms

i. The publication: each federates records, using RTI, which objects it will represent.

ii. The subscription: each federates records which attributes and interactions it needs in order to execute its tasks.

4) The VRML: The VRML [72] is a standard used to describe arbitrary three-dimensional scenes or objects [35]. This file format not only allows the description of the geometry and materials of any 3D structure, but also represents the animations and behaviors through complex event routing and behavior scripting [35]. VRML is organized around the concepts of nodes. In fact, the scene graph is composed of a hierarchy of nodes where some nodes are parents that contain other nodes known as leaf nodes. One of the advantages of using VRML is that the VRML world can use resources available in the internet, and this due to the fact that the nodes are defined through a URL.

VRML allows animating 3D scenes. Thus an event propagation system is standardized. VRML allows also writing applications and tying them to the scene graph. To this end, VRML allows scripts to be connected with parts of the scene graph. These scripts are

either defined using JAVA or ECMA script [35]. The event system activates the scripts which are executed and provided with a standardized interface to change the scene graph.

5) MPEG: The Moving Picture Experts Group (MPEG) [73] is the standard for the coding of audio-visual data. MPEG has standardized the following compression formats:

i. MPEG-4: is a standard used specially to compress audio and visual digital data. MPEG used VRML as a mechanism to compose the scene and to extend it to support 2D objects and 3D mesh coding [35]

ii. MPEG-7: is a multimedia content description standard. The description is combined with the content itself, to permit a fast searching for material that is for interest to the user. It uses XML to store metadata and can be attached to timecode in order to tag particular events.

iii. MPEG-21: aims at defining an open framework for multimedia applications. MPEG21 is based on two concepts: the definition of a fundamental unit of distribution and transaction, and the concept of users interacting with them.

2.4.4. CVEs Architectures

As mentioned before, we have known a growing interest for CVEs and C-HAVE class of applications. However, it is not always easy to build a Collaborative virtual environment. One of the most challenging issues is the choice of the architecture. Most of the designed applications have chosen one of the following architecture: peer-to-peer or client-server architecture.

1) Client-Server Model: In the past decade, the Client Server model was the first choice of several networked games and Collaborative Virtual Environment applications [18]. In

this model, the server (a main node) controls the virtual environment's management. The server stores the information of the different objects and elements of the virtual environment and keeps a list of all the clients connected to it. When an object moves, the server updates the object's state and delivers the information to the clients. The server is also responsible for routing information to the clients connected to it. If a client wants to access an object and process it, it has to ask the server first. When the process is finished, the client sends the object's new state to the server, who transfers it to the concerned clients. The network packets will then be transferred twice: from the client to the server and from the server to the other clients.

The client server model, illustrated in Figure 2.6 (A), was successfully used in many applications. However, in large-scale simulations, this model shows to have many drawbacks [18], such as important delay, latency, and packet loss. Moreover, it has been shown that one server can not handle the virtual environment by itself and multiple servers' solution was brought. However, this solution is not cost effective as pointed out in [35].

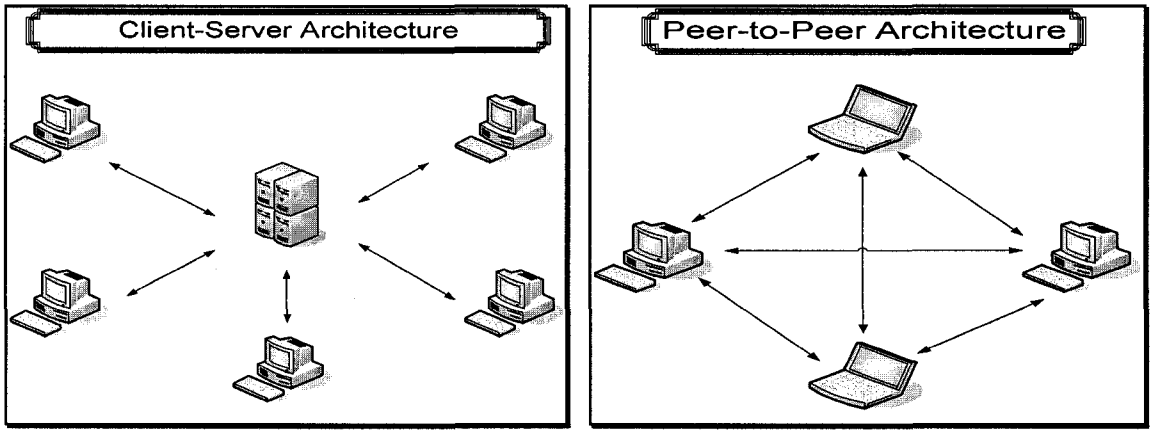


FIGURE 2.6: CLIENT SERVER MODEL (A), PEER-TO-PEER MODEL (B)

2) Peer-To-Peer Model: In the peer-to-peer model, illustrated in Figure 2.6 (B), nodes communicate directly with each other and therefore do not need a main node to exchange data. When a client transfers a message, it sends it to all the participants. This message is not duplicated, but sent via the use of reliable multicast transport protocols and travels only once to reach directly the desired clients. As opposed to the client server model, network delays are decreased [35]. This characteristic was beneficial for collaborative virtual environment applications since the delay is decreased, and also helped the applications to scale. For this reason, peer-to-peer model has been the first choice for several large scale applications [35]. However, peer-to-peer approach presents many drawbacks [35]. In fact, this model does not address the concurrency and causality of actions. Moreover, when a message is sent, it is delivered to all the clients, despite those who are not interested in receiving the message, since it is so difficult to send the message to specific clients.

2.4.5. CVEs Applications

The research on the CVEs has been gaining increasing interest from a technological perspective and the applications using CVEs have been growing. These applications either aim to develop learning and training environments for different scenarios such as Tele-surgery training, military training, and industrial training, or have entrainment purposes and are used in gaming scenarios.

1) Tele-Surgery Applications: Various Tele-surgery applications have been designed using the CVEs. Most of these applications utilize haptic devices in order to give a sense of touch and realism to the users. Tele-surgery applications aim to train medical students, and surgeons on performing surgeries on virtual patients. This allows them to perform the

same surgery many times and to be well trained for it. Nowadays, surgeons have to travel to perform a surgery on a patient. In order to minimize the costs of the trip and to save time, surgeons will be able, in the future, to be Tele-present and to perform surgeries on real patients at a distance. A successful Tele-surgery application has to satisfy the following requirement, i.e. reliability, acceptable end-to-end delay, and low data error rates. Tele-Surgery applications include:

i. A *Cataract Tele-Surgery Training Application* [36]: has been designed in the DISCOVER Research Laboratory at the University of Ottawa. It is a hapto-visual application that aims to train surgeons on performing a cataract surgery and to learn the surgical procedure. The collaboration is done over the CANARIE CA*net4 photonic network. El-Far *et al.* explained in [36] that the cataract Tele-surgery has been performed using two users geographically distant and having haptic devices at both ends.

ii. A *Tracheotomy Tele-Surgery Application*: has been also designed in the DISCOVER Research Laboratory at the University of Ottawa. It is an application that purposes to teach surgeons on performing the tracheotomy surgery using haptic devices such as the PHANToM [61].

iii. A *Brain Tumor Tele-Surgery Application*: has been designed at the PARADISE Research Laboratory at the University of Ottawa. This Tele-surgery is distributed and collaborative, and targets to allow surgeons to learn how to accomplish a brain tumor surgery. Both users are using haptic devices such as the PHANToM [61].

2) Military Training: The first military training application was designed by the NASA Ames Research Center during the mid 1980. This application used the first virtual reality HMD [56], and Heads-Up Displays [80], in order to improve the augmented reality and night vision [57].

The number of applications targeting military training is increasing in an unbelievable way. These applications are designed to provide real world combat scenarios. Piloting students having this training feel that it is a real mission due to the technology used.

The Military Training Technology Online Edition [54] explains that the training is being implemented using new PMATS (Predator Aircrew Mission Training System) simulators. According to some navy soldiers [54], these training are very important and needed. Students that receive this training are taught on the Unmanned Aerial Vehicles (UAV)'s Infrared and visual cameras and video equipment. This includes the use of simulated images from electro-optical, infrared and synthetic aperture radar sensors which includes varied terrain, weather conditions and lighting [54]. From its side, the Simulation Learning Company 3DSolve [55] has been selected to join the American Army in order to participate in creating a military training simulation scenarios. The military training applications consist on 3D simulations with real scenarios. Students use haptic devices such as the Head Mounted Display (HMD) [56] or the Data Glove [28]. These devices allow the student to experience real world combat scenarios.

These simulations help soldiers to better understand and improve the way they deal with situations and the enemy during the war.

3) Industrial Training: Several applications have been designed targeting industrial training. In fact, in order to cut down the expenses, many industries are using the CVEs when training engineers [37]. J.C Oliveira *et al.* presented in [37] a prototype of an industrial training application that aims to allow user to learn how to repair a faulty ATM switch. This application allows the user to examine the operation of the switch and its cards. When he finds the faulty card, the user is able to remove it, replace it by a new card in the switch, and put the faulty card in the repair table [37]. The same prototype allows the user to watch a video demonstrating the correct procedure when executing some actions. Mechanical engineers have the ability to design and test the arrangements of new components [35]. In fact, in order to reduce the costs of creating a new car, mechanical engineers prefer to design a virtual prototype and to test all the components. Similarly, architects are using virtual prototypes of the buildings they are designing, in order to have a sense of space and realism [35].

4) Entertainment: Many applications having entertainment purposes have been designed in the last decade. This sector knows an important increase nowadays and the use of CVEs in online-gaming knows a growing interest. Online gaming applications can support many users simultaneously (hundreds or even more), who play inside a shared 3D world via the internet. Most of the online gaming is based on client-server architecture [35]. In order to ensure the scalability, this architecture uses a cluster server. These online gaming applications are divided into two groups:

i. Online-gaming applications that have well defined goals. This category allows users to play and meet the goals defined by the game. Several games belong to this category, such as Lineage II, Everquest II [38].

ii. Massively Multiplayer Online Social Games (MMOSG): This category emphasizes on the socialization and the improvement of the virtual world. Moreover, MMOSG can be used in other domains such as education and business [38]. MMOSG include Second Life and Active Worlds.

Several entertainment applications are audio-visual and use haptic devices, such as the Head Mounted Display (HMD) [56] that allows the user to have a sense of touch and realism.

All of these applications have distributed geographically users that collaborate in real time in closely coupled and highly synchronized tasks to manipulate shared objects. For a successful execution, these applications have to satisfy the CVEs requirements which are the *reliability, minimum delay, scalability*, and *synchronization.*

Nowadays there is a growing interest in Collaborative Virtual Environments based Class of applications. In order to ensure collaboration between remote distributed users, these applications have to meet the requirement described above. To that end, several protocols have been designed to transfer data between remote and distributed users and to ensure the satisfaction of the above requirement. The following Section 2.5 describes the protocols that were designed.

2.5. Protocols Designed For CVEs Class of Applications

The research on the CVE has received a great deal of attention in recent years, and we are witnessing a large number of applications using CVE technology. These applications include Tele-Presence, Tele-Medicine, Tele-Learning, industrial design, and gaming, just to mention a few.

By using the haptic devices in the CVE, the degree of synchronization has become higher [17]. In fact, delays can easily make the haptic interface instable [33], and, thus, affect the collaboration, the coordination of the users' actions. Therefore, many protocols were designed in order to reduce delays and jitters when transmitting information for shared virtual simulations over large networks. Other protocols have also been proposed in the literature for a variety of CVE applications. All these proposed protocols can be classified into three categories:

1) Protocols dealing with networking problems,
2) Protocols based upon a Human-Computer Interaction, and
3) Protocols that use predictive techniques.

2.5.1. Protocols Dealing With Networking Problems

For networking related problems, many protocols have been designed in order to solve network problems such as the packet delay, loss, and jitter.

(1) The Selective Reliable Transmission Protocol (SRTP)

The SRTP protocol was designed by M. Pullen *et al.* at the George Mason University to support large-scale Distributed Interactive Simulations (DIS) [35] and Distributed Virtual Simulation (DVS). SRTP has been successfully used in the Run Time Infrastructure

(RTI) / High Level Architecture (HLA) to provide efficient real-time networking for distributed simulation [2]. It occurs at a level directly above a network protocol [1, 2]. SRTP reduces the network capacity requirement by transmitting only those data elements [2], such as position, that change frequently, by using real-time best-effort multicast. For the data that change rarely; or not at all; SRTP transmits them reliably one time. In order to provide reliable transmission of data that change rarely, SRTP provides a standardized set of communication services than can be combined with User Datagram Protocol (UDP) [2].

Depending on the type of data delivered SRTP has the following three (3) modes of operations:

i. ***Mode Zero***: for transient data that change frequently, such as position. This kind of data does not require reliability [1, 2, 3, and 4]. For the mode 0, SRTP uses a multicast architecture that operates as pure best-effort services.

ii. ***Mode One:*** for data that must be received reliably by all members who are in the same multicast group. This data can be a reference data in an object state protocol. In this mode, it is important to maintain low end-to-end delay. SRTP maintains the reliability by including information in the Mode 0 message to indicate when the Mode 1 update occurred. By checking the sequence number that is in the header of the packet, the receiver is able to detect a lost packet and to multicast a NACK to all the members to inform them of the packet missed [1, 2, and 3]. Only the most recent mode 1 data of each entity stream is retransmitted in case of a lost packet [1]. For this mode, SRTP uses a NACK-

based reliable approach, with a "NACK suppression" mechanism as an effort to avoid the NACK implosion problem.

iii. ***Mode Two***: for a transaction with a single dynamically-determined receiver in the multicast group [1, 2, 3, and 4]. For this mode, SRTP ensures a reliable and timely transport. It uses a positive acknowledgement for data that must be received reliably by a single known member in the group. SRTP uses either Transmission Control Protocol (TCP) or ACK-based multicast to ensure the reliability. When the MODE2 data is received by the proper node, it multicasts an ACK to all the members of the group. These ACK are ignored by all the nodes for which they are not intended [3].

Modes 0 and 1 were designed in order to transmit the state of an entity, while mode 2 was designed for the transmission of dynamic interactions between pairs of entities [4].

SRTP supports multiple streams of entity state information and provides an ID field that allows to associate data belonging to the same entity stream [1]. In order to keep the ordering, SRTP uses the sequence numbers within each entity state stream [1].

The congestion control plan is very important in SRTP, because even a moderate amount of congestion can destroy the real-time performance of this protocol [4]. The congestion control is achieved for the mode 2, since TCP detects the congestion level in the network and reduces the transmission rate of the data. For the multicast case, Mode 0 and Mode 1, the congestion control is maintained using the NACK Suppression technique [1]. In fact, for Mode 1, receivers do not use ACK messages to inform the sender that the message was received, but they use NACK messages instead when a lost packet is detected. Even NACKs can still produce an implosion at the sender [1, 2], and the solution applied for

this issue is the NACK Suppression mechanism, in which every NACK message is delayed a random period of time in order to distribute the transmission of NACKs back to the sender; and a NACK message scheduled to be sent is cancelled in case a NACK for the same data is sent from another host [1]. The drawback of this approach is that the NACKs messages have to be sent to the entire multicast group instead of a unicast to the sender [1]. However, the main problem that SRTP encounters is that it does not address the synchronous collaboration because of its NACK based approach [9].

(2) The Synchronous Collaboration Transport Protocol (SCTP)

SCTP is a host-to-host layer protocol especially used to ensure the collaboration among users. It is scalable and fast due to the utilization of IP multicast [8]. SCTP is based on the notion of an *Interaction Stream*, which means that a series of update messages, related to a sequence of interactions of a user with a shared object, is grouped into one Stream [9]. This stream is composed of two types of messages: *key-update* messages and *regular update* messages. The update messages represent the position or the motion of an object, and are sent through the best effort transport. The key update message represents the last update message sent and has to be delivered reliably. An SCTP packet format is illustrated in Figure 2.7.

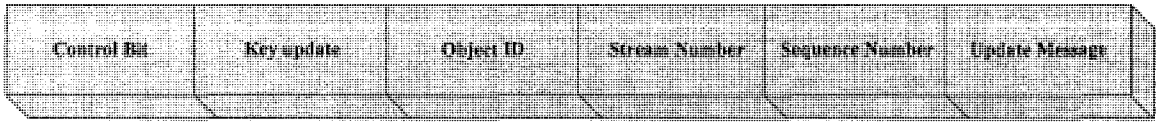


FIGURE 2.7: SCTP PACKET FORMAT

The *Key Update* field informs the receiver if the received message is a key update or not. From the packet's header, the receiver can identify which shared object is being interacted with by looking to the *Object ID* field. The packet also provides *Stream ID* and *Sequence Number* information. *Stream ID* field indicates the ID of the current interaction stream for this object, and the *Sequence Number* field indicates the position of this specific update message in the current stream [9]. These fields allow the receiver to discard the late messages.

To ensure the reliability for the key update messages [8, 9], SCTP uses an ACK-based architecture. This means that the sender requests that these messages have to be immediately acknowledged by the receiver. In order to avoid the congestion of the protocol, the NACK based technique is used. In fact, when a packet is lost, the receiver first sends an NACK message [8] to inform that it did not receive the packet then it asks then for a re-transmission. SCTP significantly improves the users' ability to collaborate on a network where delay and packet loss is evident. However, SCTP is unable to solve the problem of jitter and cascade packet loss that it encountered and it must deal with the ACK implosion problem [1, 2, and 3]. Indeed, in order to realize reliability, SCTP uses only the ACK-based approach which may lead, in many cases, to the inconvenient ACK implosion problem.

(3) Smoothed Synchronous Collaboration Transport Protocol (S-SCTP)

Smoothed SCTP (S-SCTP) was developed in order to address the delay and the jitter problems that SCTP encounters [18, 21].

S-SCTP is an enhanced version of SCTP that adds two features:

i. On the sender side, a timestamp is added to the update message. Dodeller *et al.*

assumed that all the computers involved have a synchronized clock.

ii. Buffer is implemented on the receiver side in order to make the jitter smooth [18].

Figure 2.8 illustrates a Smoothed SCTP packet format.

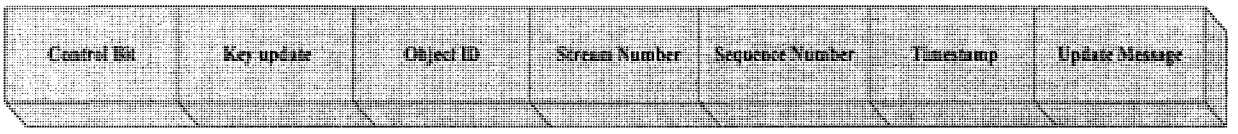


FIGURE 2.8: SMOOTHED SCTP PACKET FORMAT

When updates reach the receiver side, they are put in a bucket according to their timestamp for when they were sent. Periodically, the receiver checks the bucket corresponding to buckets that have been sent ψ milliseconds ago [21]. If there are any update messages corresponding to that period of time, then the receiver forwards them to the application [21]. This allows for the processing of all update messages, even those that have been generated locally, with a fixed delay of ψ that is higher than the actual network delay but constant.

Smoothed SCTP fixed the jitter problem but could not deal with lost update messages or update messages that are delayed longer than the buffer's length of time [17].

2.5.2. Protocols Based upon Human-Computer Interaction

From a Human-Computer Interaction perspective, visual ornaments have been designed in order to visualize the network state and more specifically the delay. A study was performed by Gutwin *et al.* [34] in order to first show the nature, magnitude, and consequences of the delay and then inform the participants about the delay so that they can adopt their own natural coping strategies. Recall that the network delay combines the transmission delay, queuing delay, and retransmission delay. Many protocols were designed in order to minimize the network delay. While the protocol proposed by Gutwin *et al.* [34] deals with the delay

and reveals its magnitude to the users only to make them aware of the minimum time to wait before they can see the actions they performed, it does not minimize the network delay. This approach is based on visual ornaments, decorators, which are added to an object's representation in the user interface. These Decorators are used to present visual feedback about the presence, magnitude and effects of delay [17, 18, 19, and 34]. Gutwin *et al.* [34] proposed two families of decorators: *magnitude of delay* decorators and *past and future state continuum* decorators. The first family, *magnitude of delay decorators,* is composed of four different categories:

i. *Roundtrip time decorator*: represents the time to expect before observing the result of an action performed

ii. *Jitter decorator*: represents the variation in delay over time. This category helps predict the objects movements.

iii. *One-way temporal distance*: is a decorator that reveals the magnitude of the delay in one way, which separates the user from the object or vice-versa. Two decorators are thus used to illustrate the magnitude of the delay, one in each direction.

iv. *Third-party delay decorators*: offers, to the user, an inter-subjective view of the delay between two remote objects. This category is an important factor because it reveals, to the user, the time that an object's message takes to reach the other.

The second family of decorators proposed by Gutwin *et al.* [34] presents the predicted state of an object, based on his past states, the knowledge of constraints on his movement, and information about the ongoing delay. The purpose of this family of decorators is to assist the

users predicting the behavior of an object, and plan their activities in advance by either slowing down or waiting longer. Decorators can present some drawbacks, when used in CVEs. In fact, decorators are visual feedbacks, and they add thus visual information, which may distract the user [34]. Moreover, it is hard to receive information about the delay from the system.

2.5.3. Predictive Based Techniques

In terms of predictive techniques, an updated version of the decorator technique was designed in [17]. In conjunction with both a decorator and a networking-level solution, a prediction technique was added. In fact, Boukerche *et al.* [17] proposed to use the Smoothed SCTP, as a networking-level solution, and combine it with a decorator solution and prediction algorithm. Smooth SCTP is an enhanced version of SCTP and uses the Interaction Stream concept which has two types of messages: regular update messages and key-update messages. The S-SCTP adds a timestamps in the header of the message in order to keep a temporal order.

The predictive techniques proposed by Boukerche *et al.* [17] works as follow: All update messages generated by the system are sent to a buffer *history*. From the *sequence number* field included in the update message, we can detect a delayed or lost message in the Interaction Stream. Each time a message is delayed, the system predicts an update message and keeps it in the *history* buffer until the enforced delay time times out. If the message is not received within that time, the user is informed about the delay via the object's color's change. In fact, when the delay is detected, the predictor unit sends a message to the decorator in order to allow it to change the color. A linear prediction algorithm is then applied in order to replace the lost messages [17].

In his proposed predictive technique, Boukerche *et al.* uses three types of decorators:

i. Jitter decorator: the system accumulates the timestamps of successive update messages and calculates the delay variation or the jitter and changes thus the object's color.

ii. Direction decorator: the direction of an object is derived from its consecutive states.

iii. Trajectory decorator: is a combination of the past states of an object and its predicted future states. The user can predict the trajectory of an object based on its past states.

Boukerche *et al.* [17] claims that the proposed prediction technique ameliorated the collaboration between users and gave the best results when compared to other protocols

2.6. Summary

Collaborative Virtual Environments (CVEs) are distributed virtual reality systems that allow remote and geographically distributed users to collaborate in closely and coupled highly synchronized tasks to manipulate a shared virtual object. By bringing the notion of "Haptic" to the CVEs, the manipulation of the shared virtual object has improved. All of the CVE applications have to achieve collaboration by satisfying the CVE requirement, i.e.: synchronization, minimum delay, scalability and reliability. They also must have the following components:

i. Shared virtual environment

ii. A consistency method

iii. A computer graphic rendering program

iv. A haptic device interface

Several applications have been designed. They either target learning and training perspectives such as tele-surgery training [36], military training [57], and industrial training [37], or have entrainment purposes and are used in gaming scenarios [38].

CVE applications have known a growing interest and the need to unify standards has become urgent. These standards include: RTP/I [39], DIS [74], HLA [75], VRML [72], and MPEG [73]. All the applications designed have either chosen client-server or peer-to-peer architectures. Many protocols have been created in order to achieve collaboration. These techniques can be classified into the following three categories:

i. Protocols dealing with network problems: SRTP [1,2], SCTP [8,9], and S-SCTP [21]

ii. Protocols having human-computer interaction [34]

iii. Protocols using predictive techniques [17]

To the best of our knowledge, none of the techniques proposed, so far, were able to meet the CVE requirement. As a consequence, the design of an efficient transport protocol to meet these requirements will have a profound impact for CVE class of applications. As previously noted in Section 2.5.1, packets that are exchanged by participants have to reach several destinations. For this reason, the multicast transport is preferred when compared to traditional unicast models.

The next chapter introduces multicast transport presenting a detailed description of its features.

Chapter 3

Multicast Transport Protocols

3.1. Introduction

The aim of this chapter is to give an overview of the multicast transport protocols designed. The first section gives an introduction to the multicasting and the features that can be added to satisfy each application's requirement, while the second section describes the multicast protocols by classifying them according to several features.

Collaborative Virtual Environments require that geographically distributed users collaborate in closely coupled and highly synchronized tasks to manipulate shared objects. In order to keep the users updated with the changes, and to maintain the synchronization between them, an efficient transport protocol to deliver data is needed. The traditional unicast model that Transport Control Protocol (TCP) [51] offers is inefficient for such applications. In fact, it is not useful to transmit the same data to each receiver across the network. CVE applications use large scale distributed simulations where hundreds or even thousands of distributed users intercommunicate in real time. For this reason, it is necessary to use the multicast approach to deliver data.

Several multicast protocols have been proposed by many researchers in the last decade. This interest was mainly due to the numerous communication applications, such as multipoint data dissemination or conferencing tools, which use multicast as a solution to deliver information to the different users of the application.

3.2. Definition and Features

3.2.1. Definition

Multicast is being chosen more often by several communication applications as a solution to deliver data to a large number of receivers simultaneously. During the last decade, research on multicasting received a great deal of interest. Many definitions of multicasting have been given in the literature. Multicasting [78] can be defined as:

> *"Multicast is the delivery of information to a group of destinations simultaneously using the most efficient strategy to deliver the messages over each link of the network only once, creating copies only when the links to the destinations split."*

Sahasrabuddhe *et al.* [53] also defined multicasting as follow:

> *"Multicasting is the ability of a communication network to accept a single message from an application and to deliver copies of the message to multiple recipients at different locations."*

B.A. Forouzan [52], in his book *TCP/IP Protocol Suite*, has defined multicasting as follow:

> *"In multicasting routing, there is one source and a group of destinations. The relationship is one-to-many. In this type of communication, the source address is*

a unicast address, but the destination address is a group address. The group address defines the members of the group."

Multicasting requires less bandwidth and delays than multiple unicasting [52]. In fact, the sender in multicasting sends only one packet that is duplicated by the routers. However, in multiple unicasting, several packets are sent from the source, which requires more bandwidth to handle all of the messages sent. Multicasting has been then proven to be the best delivery solution for many applications.

During the last years, several protocols, mechanisms, and techniques were designed in order to ensure multicasting. Multicast transport protocols have been the subject of an active research since. This research has been specially supported by both the Internet Engineering and Internet Research Task Forces (IETF and IRTF) that were coordinating the development and standardization of multicast transport protocols [40]. To provide multicast services, a large number of protocols have been designed. However, these protocols were designated to satisfy the requirement of special class of applications, and were then applied in different environments. Among these applications, we mention: multimedia conferencing systems, distributed interactive simulations [35], chat groups, distance learning [23], media applications, multi-player games [38], distributed whiteboard tools, collaborative applications, just to mention a few.

Each proposed protocol has its own features to satisfy the requirement of its application: Each protocol has its own technique to ensure the reliability, quality of service (QoS), error recovery, ordering, congestion control, and flow control. Since there are several applications, then requirement and environments may differ from one application to another one. As a consequence, the design of a multicast protocol that satisfies all the

requirements is a challenging problem and most likely impossible. In the following section, we will present the features of the multicast protocols.

3.2.2. Features

A number of multicast protocols have been designed in order to address a variety of problems for different type of applications [40]. Most of these protocols have to acquire certain features in order to satisfy the application's requirement. In the following, we will present and discuss the different features that most of the multicast protocols designed need.

1) Reliability: A protocol is said "reliable" if it guarantees that data transmitted by the sender is delivered to the intended receivers in order and without duplication. Both the sender and receiver can test if the reliability is satisfied. For a unicast transmission, the reliability is easy to satisfy since there is only one receiver that either receives or does not receive all the data transmitted by the sender. However, for a multicast transmission, it is not always the case, and protocols need strong reliability properties, to be able to deliver data to the intended receivers.

There are many techniques to ensure the reliability. Usually, unicast transport protocols such as Transport Control Protocol (TCP) use ACK messages to ensure the reliability and to detect packet losses: Receivers have to send ACK messages to the sender to inform it about the packet received. By checking the packet's sequence number, the sender can detect a lost packet. This approach is referred to as *sender-initiated,* since the sender is responsible for detecting lost packets. However, this approach is not efficient for multicast protocols. In fact, if each receiver sends an ACK message, then this may create an ACK implosion problem.

Another approach commonly used is the *receiver-initiated*, where it is the responsibility of the receiver to detect packet losses. This approach is used in order to solve the ACK implosion problem. Receivers have to check the packet's sequence number when they receive messages. If a missing packet is detected, the receiver has to send a NACK message to the sender, to inform it about the missing packet.

Most of the multicast transport protocols use the receiver-initiated technique, and a study done by *Pingali et al.* [49] shows that the receiver-initiated approach performs better than the sender-initiated one. In order to ensure the reliability, and depending on the requirement, some protocols, such as RMP or RBP, described in next sections, combine both approaches.

2) Congestion and Flow Control: Congestion control deals with controlling the traffic in the network, by reducing, most of the time, the rate of sending packets. Congestion control is important for a wide number of applications [42]. For this reason, many techniques and algorithms are used to control the congestion. Protocols control the congestion by checking the type and the amount of feedback received from the network. This feedback is composed especially of the delay and loss of packets. In order to avoid congestion, multicast protocols either require that receivers maintain the same speed [42], or allow certain receivers to bring down their rate requirement when they detect congestion.

3) Ordering: Several applications require that packets are received in order because out-of order pacekets may lead receivers to a different state. To that end, many protocols are

designed in a way to provide a mechanism to ensure complete ordering. This feature is realized via the use of sequence numbers. Ordering is a very important feature for multicast protocols. In fact, ordering allows receivers to have the same and correct execution. It also helps them to detect duplicate messages and discard them when it is necessary. There are different types of ordering [6, 42]

i. Total order means that all of the receivers have the same order of packets

ii. Causal ordering means that if sending a message m causally precedes sending message m' then no receiver delivers the message m' before the message m.

Collaborative applications necessitate that messages are totally ordered delivered [40].

4) Error Recovery: An error recovery mechanism is very important for reliable multicast protocols. Many approaches and algorithms exist in order to select a repair server to perform error recovery. If a single server is used to ensure the responsibility of error recovery, then this leads to the well-known implosion problem [50] caused by the ACK and NACK messages sent by the receivers. For this reason, some protocols use the receiver based reliability where the receiver is responsible for detecting lost packets and asking for the repair. In some protocols, such as Scalable Reliable multicast (SRM) [10] described later, every user can participate in the repair process if it holds the repair message. In fact, when a member detects a lost packet, it multicasts a retransmission request for the entire group. Any user that has the correct message can multicast the repair to the entire group. In order to avoid duplicate requests and repairs, a back-off algorithm is used. In some other protocols, tree-based protocols such as the Reliable Multicast Transport Protocol (RMTP) [7] described later, it is the responsibility of the

group's master to recover an error and retransmit the missing packet. In fact, in this class of protocols, receivers are grouped into local groups based on their geographic position. A designated receiver is selected in each group and is responsible for ensuring error recovery for all the receivers in its group. By using this technique, multicast protocols avoid the implosion problem at the sender side. However, since the error recovery is only concentrated on one server, this class of protocols still suffer from several problems [50]. In fact, tree-based protocols can still suffer from implosion problems, since all the requests for a retransmission in a specific group, are sent to the same designated receiver. Moreover, if the server that is responsible for the error recovery fails, then receivers can not get the repair until a new designated receiver is elected [50].

5) Group Management: Some multicast protocols do not need knowledge of group membership. The sender does not keep a track of the receivers in the multicast group. Receivers can join and leave the application without informing the sender. In this case, the sender just transmits its data to a multicast group address and receivers join to get the data [40]. This technique is specially based on IP multicast, and is used by many protocols because it facilitates the scalability [40]. However, other protocols require the knowledge of group membership. In this case, group management is either controlled by the sender or hierarchically by a designated receiver or a group master [42]. Receivers have to make the responsible of the group management aware of their status and inform it before leaving or joining a session.

Most of the multicast transport protocols ensure the scalability. In fact, the multicast transport of data is a good solution for applications that need to scale, since they do not keep a track of users in the group: the sender just multicast the message and receivers join the group and get the message [40].

In the next session, we will classify the multicast transport protocols designed in the literature and describe the most important protocols widely used.

3.3. Classification of the Multicast Protocols

Several transport protocols were designed in the literature [40, 42]. However, not all of these protocols were designed in order to be applied in the same class of applications and to satisfy the same objective. In her survey "Multicast Transport Protocols: A Survey and Taxonomy" [40], Katia Obraczka classified the multicast transport protocols as follow:

i. General purpose protocols

ii. Multicast Interactive Applications

iii. Data Distribution Services

The classification of the multicast transport protocols is illustrated in Figure 3.9. Most of the Multicast transport protocols designed has been for a specific class of applications: Interactive Applications and Data Distribution services. However some of the techniques designed earlier did not target a specific class of application and they are therefore for a general purpose. This classification is elaborated in the next section.

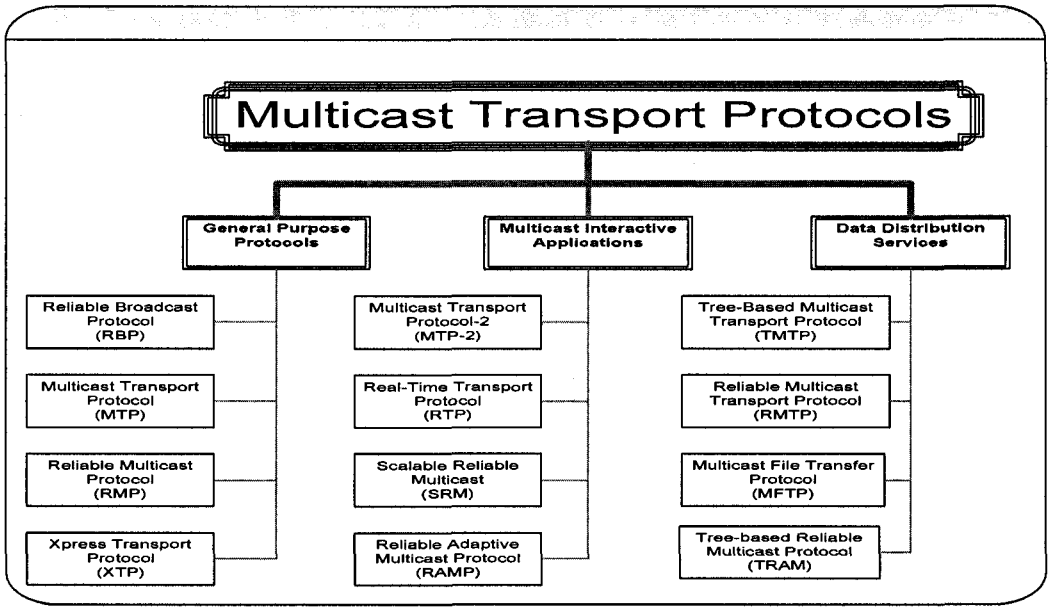


FIGURE 3.9: CLASSIFICATION OF MULTICAST TRANSPORT PROTOCOLS

3.3.1. General Purpose Protocols

This category represents protocols that have been designed at the early stage. These protocols were not conceived for a specific class of application and they are therefore under general purpose class of protocols. These protocols are all message-oriented. Among protocols designed in this class, we mention: Reliable Broadcast Protocol (RBP), Multicast Transport Protocol (MTP), Reliable Multicast Protocol (RMP), and Xpress Transport Protocol (XTP).

1) Reliable Broadcast Protocol (RBP): This protocol [40] was designed in the early eighties. It offers multipoint communication between sites connected by a local area broadcast network [40]. In order to broadcast all the messages to receivers, the sender transmits the message to a primary receiver called a token site. The primary receiver gets and stores the message transmitted by the source that continues to retransmit the broadcast message until it gets an ACK message from the token site confirming that it

receives the complete message. This ACK message contains a global timestamp that the receivers use to order and find a lost message. The token site sends the message to the broadcast group. When a receiver detects a lost message, it sends a NACK message to the token site, which retransmits the missing message. The role of token site rotates among receivers in the broadcast group in order to reduce the storage costs. A member can be a token site only if it has all the messages sent earlier than the timestamp of the current token site. In order to ensure the reliability and ordering, RBP mixes the Negative Acknowledgement (NACK) with the positive acknowledgment (ACK).

2) Multicast Transport Protocol (MTP): This protocol [69] was influenced by RBP [40]. It offers reliable and ordered delivery of data [40]. As described in RBP, MTP has a group master, which ensures that the data is delivered reliably and in order. In addition to the reliable and ordered delivery of data, MTP ensures the synchronization among the receivers to agree on the order of receipt of all messages [41]. Users in this protocol can have one of these roles:

i. The group master: it ensures that the data is delivered reliably and in order. It gives the token to the member who wants to transmit data. Moreover, it checks the membership and performance of the group members. In fact, if a group member that has the token disconnects, then the master removes it from the group and rejects the member's outstanding messages.

ii. The producer: it is allowed to send messages to the entire group and to consume data sent from other producers. Before transmitting the message, the producer has to get the token from the group master.

iii. The consumer: it is allowed to only receive messages from the sender.

If a new member wants to join the group, it has to send an adhesion message to the group master, which will decide if the new member can join the group. Before allowing the new member to join the group, the group master has to have all the transmit tokens in order to ensure that the new member gets only the complete messages. The group master informs then its client application about the new member. When a lost message is detected, MTP uses the NACK messages. In fact, as soon as a consumer detects a lost message, it sends a NACK to the message's producer that multicasts the missing message to the entire group [40]. In terms of flow control, MTP is based on a fixed size transmission window, which defines the maximum number of data packets that the member can send to the group within a period of time. Every member of the group has to accept the actual window size when it joins the group [40].

3) Reliable Multicast Protocol (RMP): This protocol [76] offers a totally ordered and reliable multipoint transport on top of an unreliable multicast service such as [40, 42] IP multicasting. Note that RMP [40, 42] is inspired from RBP [40].

The flow and congestion control mechanisms for RMP are based on the algorithms Van Jacobson that were designed for the Transfer Control Protocol (TCP). These mechanisms allow RMP to offer high performance over LANs and WANs [42]. Like in RBP, the sender transmits the message to a primary receiver a *token site*, which sends an ACK message, to the sender, to confirm that it receives the entire message. The token site, which rotates among the members of the group, is responsible of managing new members and those that want to leave the session. In order to accelerate the errors recovery, RMP

uses NACK messages. A small number of ACK messages are used to guarantee the delivery [42]. In order to implement the flow control and when it detects a lost packet, the receiver does not send a request to the token site but it multicasts a NACK to the entire group. When the sender gets the NACK message, it backs off exponentially and retransmits the missing packet.

4) Xpress Transport Protocol (XTP): XTP [42, 77], which is an enhanced version of the Xpress Transfer Protocol, is a network-and-transport-layer protocol [42]. It unifies unicast and multicast features. XTP is a one-to-many protocol, designated to serve small and medium groups. For each association established between the sender and a receiver, XTP associates a state information packet. This packet contains important information such as the sequence number, size of the transmission window, an estimated round-trip time, and rate of transfer [6]. The sending application controls the group membership, by deciding which member can join the application. To that end, the sending application uses the Multicast Group Management service (MGM) offered by XTP [40]. XTP ensures the reliable delivery of data by the use of TCP. For data that do not need to be transferred reliably, UDP is used as best effort transfer.

XTP is a sender reliable protocol. In fact, the *Loss Estimation System (LES)* is at the sender, and receivers have to send information about the packets received periodically. When a receiver detects a lost packet, it sends a NACK message to the sender, which multicasts the missing message to the entire group. The sender can choose when the receivers should produce ACK messages. Depending from the sender's choice, receivers

can either send always ACK messages, sometimes or never [40]. XTP provides both rate-and-window- based flow control.

3.3.2. Multicast Interactive Applications

Interactive application refers to a human-computer interaction application. These applications support one or more users and one or more computers. Multipoint Interactive Applications have known a growing interest in the last decade. These applications include: multi-participant multimedia conferencing services, distributed whiteboard application, circuit-switched, and gigabit network environments. Reliability in these applications is really important, however, interactive applications are willing to resign on the reliability in favor of real-time delivery. In this section, we will present the most important protocols designed to satisfy the requirement of interactive applications. Among protocols developed in this category, we mention: The Multicast Transport Protocol-2 (MTP-2), Real-Time Transport Protocol (RTP), Scalable Reliable Multicast (SRM), and Reliable Adaptive Multicast Protocol (RAMP).

1) Multicast Transport Protocol-2 (MTP-2): This protocol [82], which is an enhanced version of MTP described above, was designed by C. Bormann et al. in order to satisfy the requirement of collaborative applications [40].

The changes applied to MTP-2 are the following [40]:

i. *Immediate joins*: In MTP, if a new user wants to join a session, it has to wait until the master gets all the transmitted tokens. In MTP-2, new users can join the multicast group by ignoring all the messages sent before the join message.

ii. *Master recovery*: MTP did not study the failure of the master and did not thus propose any solution for this problem. To that end, MTP-2 studied the master's failure problem and addresses it by allowing the group members to detect the failure and to select a new master.

iii. *Dynamic group parameter adjustment*: Unlike in MTP, where the transmission window size is fixed, the MTP-2 allows the master to dynamically change the transmission size according to the network state.

iv. *Differentiated services*: MTP-2 allows the sender to choose if it needs an atomic message delivery for every message sent. If atomicity is not required, the receiver will send an atomic message when it gets the entire message.

The reliability is ensured on the receiver side. In fact, when the receiver detects a lost packet, it sends a NACK message to the sender, which multicasts the missing packet to the entire group [6].

2) Real-Time Transport Protocol (RTP): This protocol [81] was developed by the Audio-Video Transport Working Group of the IETF in 1996. RTP was designed in order to be applied in multi-participant multimedia conferencing services over the internet [40]: It delivers audio and video packets over the internet with real-time characteristics, without ensuring any reliable or order delivery. To that end, RTP applies User Datagram Protocol (UDP) [44] to deliver data and uses the sequence numbers in order to re-order the packets. RTP tries to prevent the packet loss and does not tolerate the jitter. In fact, if a packet is lost in a real time application then, there is no need to retransmit it again.

RTP uses Real Time Control Protocol (RTPC) [83] in order to control the information and monitor the quality of service (QoS). In fact, RTCP provides three types of reports exchanged between the source and receivers: the Sender report, Receiver report, and Source description [43]. These reports have important information such as the number of packets sent, number of packets lost, an estimation of the delay, and an estimation of the jitter. All this information allows the sender to have an idea about the network state and to adapt according the transmission rate.

3) Scalable Reliable Multicast (SRM): SRM [10] is a framework that has been designed in a distributed whiteboard application, which has been used on a global scale with sessions having few hundred participants [6]. This protocol is efficient [14], robust and scale well when it is used for large networks and large sessions. SRM [10] is designed in a way to deliver all the data to all the group members not including any particular delivery order. The transport method is the best effort, but the reliability is built on an end-to-end basis. For the delivery, SRM is based on the IP multicast, but it added a special feature: in fact, SRM forces all the members that want to receive the data sent to the group, to announce themselves by sending a "join" message multicasting it to all the members of the group. When a member generates new data, it sends it to the multicast group. In order to keep track of the packets received, every member has to send periodically a session message that contains the highest sequence number received from each member [10]. It also allows the users to know the current participants of the session. Each receiver is individually responsible for detecting the lost packets and this by finding a gap in the sequence space [10]. In that case, it just requests for a re-transmission of the

lost packet. It starts by sending a *negative acknowledgment (NACK)* to all members. Everyone can then participate and send the repair. This is important since the original sender does not have to make the repair process especially if the distance[1] between him and the NACK sender is large [10]. With the involvement of every member, SRM makes the repair process faster [6, 10]. The members that participate in the repair process use the slotting-damping technique. In fact, when a member requests for a re-transmission, it sends it to the multicast group. The closest member (the one who has the right packet) times out first and multicast the repair packet. The other members that were supposed to send a request, and hear the request already sent, suppress their own request and wait for the repair. Similarly, the members that heard the request, that have the right packet and scheduled the answer, suppress their repair packet from their buffer when they hear the answer of the first member. The main concern of SRM is the congestion control. In order to avoid it, SRM uses the *exponential backoff* before the NACKS and repairs. This means that the user, who is going to send a NACK or a repair, sends its packet and sets a timer t. If he does not receive an answer before the timeout, he sends again the same packet and waits for 2^t. The main issue that SRM encountered is the important delay during the transmission of the message.

4) Reliable Adaptive Multicast Protocol (RAMP): RAMP [45] was designed in order to be applied in all-optical, circuit-switched, and gigabit network environments [40]. This protocol is a transport layer protocol that runs over network layer protocols such as IP multicast [45]. RAMP [45] ensures reliable and ordered delivery of all the data sent.

---

[1] The distance is calculated using the session messages and the round-trip-time it takes for a packet to travel between two members

RAMP also offers sender based-reliability: the sender can choose to send the data either reliably or unreliably.

A session can start when the sender sends a *connect* message to an IP multicast address and that it receives an *accept* message from at least one receiver. Receiver can detect lost packet by checking the sequence numbers in the packet. When it is the case, the receiver sends (unicast) a NACK message to the sender to inform it about the missing packet. Depending on the number of NACK messages received, the sender either unicast or multicast the repair packet [40]. Depending on the transmission's rate and the receivers' number, the sender can shift between two modes: idle or burst [40]. In the burst mode, the sender requires a reliable transfer: receivers have to acknowledge the reception of the messages. If the sender does not receive an ACK message from a receiver, it keeps retrying for a certain time and deletes it from its membership list then. The sender is in the idle mode and sends idle messages, either when it accepts the first *accept* message or when it has no data to transfer. In this protocol, the sender controls the group membership information that is necessary for the setup of circuits in the circuit-switched networks. However RAMP does not scale well. In order to control the flow, RAMP [45] uses a simple technique that allows the sender to adapt the transmission's rate depending on the network state.

3.3.3. Data Distribution Services

Many applications such as news and business applications need to distribute data to their clients. If the information to be sent is the same for each customer, then dissemination of data will be easier if it is sent through multicasting. For this reason, several protocols have been designed in order to satisfy the requirement of data dissemination class of

applications. In what follows, we will describe some of the protocols that were designed in this category. Among these protocols, we mention: Tree-based Multicast Protocol (TMTP), Reliable Multicast Transport Protocol (RMTP), Multicast File Transfer Protocol (MFTP), and Tree-Based Reliable Multicast Protocol (TRAM).

1) Tree-based Multicast Transport Protocol (TMTP): TMTP [46] was designed in order to offer reliable communication for Interactive Collaborative applications such as distributed shared whiteboards, distributed games or simulations [46]. It takes advantage of the efficient best effort delivery of IP multicast to deliver data. In order to improve the flow and error control, TMTP organizes users in a hierarchical control tree and this by the use of an expanding ring search. Each sub-tree is represented by a Domain Manager that takes care of the local retransmission when its local receivers detect lost packets. This protocol does not need to have a receiver to start transmitting data. Users can join the application in the middle of the simulation. The sender uses IP multicast to transmit data to the entire multicast group. The error control is distributed among several nodes, Domain Managers [46], which lead to the participation of these nodes in the error recovery. To this end, TMTP uses a limited number of NACK messages with NACK suppression mechanism. When the receiver detects a lost packet, it multicasts a NACK message combined with NACK suppression, and when its parent receives the NACK message, it multicasts the missing packet.

TMTP also provides dynamic group membership, without assuming any knowledge of it: A session directory offers primitives [40]: to create, delete, join and leave the multicast group. In order to control the flow and during the creation of the group, members accept a

predefined transmission rate to be applied during the application. Yavatkar *et al.* claimed that, when implemented and tested, TMTP was able to scale well, reduce the load on the sender, and minimize the end-to-end latency [46].

2) Reliable Multicast Transport Protocol (RMTP): RMTP [7] can be built on top of either virtual-circuit network or datagram networks. The main characteristic of this protocol is its architecture. In fact, RMTP is based on a hierarchical structure forming a multicast tree, with the sender as the root node and the receivers as the leaf nodes [7]. In order to have the tree, receivers are grouped into local regions or domains based on their proximity in the network. In each domain there is a special receiver (representative) called a Designated Receiver (DR) who is responsible for [7]: Sending acknowledgments periodically to the sender, processing acknowledgements from receivers in its domain and retransmitting lost packets to the corresponding receivers. RMTP is composed of three components: The master that controls the sessions and the communications, the DRs which can be the senders and the receivers, and finally the receivers that are only capable of receiving the messages sent by the DRs. The receivers choose their representative DR: In fact, each sender or DR sends a special packet called the SEND_ACK_TOME that contains a value of the Time to Live (TTL). The receiver will then check the TTL of each packet and choose the one that has the largest value of TTL. This chosen node is going to be the DR of that receiver.

RMTP also supports multi-level hierarchy of local regions [7]. In that case, the sender will receive only as many status messages as there are DRs in the highest level of the multicast tree. RMTP assumes that there is a Session Manager who takes care of the

connection parameters and provides them to the sender and the receivers. The connection parameters are the Packet Size, the interval T_{send} and Ws, the number of packets transmitted, which varies depending on the network state. This Session Manager is not a part of RMTP but is used to manage the RMTP session.

The protocol works as follow [7]: The sender sends the message to all the DRs (of the highest level) who will be responsible of transmitting the data to their local receivers. The message transferred by the sender is divided into fixed-size data packets with the exception of the last one. The first packets are referred to as DATA but the last one is called DATA_EOF. For each packet DATA correspond a sequence number that allows the receivers to order the packets and to detect if there is any lost packet [7]. The receivers have to send periodically their status packets to their DRs indicating which packets have been successfully received, in order to enable them to check if there is a lost packet. In this case, if the number of the receivers asking for a retransmission exceeds a certain threshold, then the lost packet is multicast; otherwise DRs just perform and end-to end transmission of the right packet to the desired receiver and reduce by that *the end-to-end delay* significantly. The DRs, from their side, will send their own status to the sender indicating which packets they have received and which packets they have not received. By using this technique, there is only one ACK message sent to the original sender per local region. This prevents the problem of ACK implosion. By reducing the unnecessary retransmission that the original sender has to send, in case of a lost packet, this protocol achieves *high throughput* and a *low end-to-end delay* [7].

In order to avoid the network congestion, the source sends the data at a variable rate and this based on the feedback from the group [11]. RMTP is designed in a way to facilitate

the scalability [7], and based on [14], RMTP has been shown via an analysis model to be the most scalable choice. RMTP is based on the IP-Multicast technique. In fact, the sender does not have a track of the receivers and this gives *a high degree of scalability.* Receivers can join and leave the session when they want to without informing the sender. Since RMTP allows users to join the group at any time, it has two features that allow the late users to catch up with the rest [7].

i. The *immediate transmission request*: when the user finds out that he joined the group in the middle of the transmission, he requests for all the packets that he missed. He sends to his DR a special packet ACK_TXNOW asking for the missing packets. The DR sends these packets using unicast.

ii. The *data cache in the sender* and the *DRs*. In fact, the DRs and the sender should have a buffer that contains the message sent during the session. This feature allows the receivers to request for a retransmission of a lost packet from the corresponding DR or Sender.

In order to provide a reliable delivery to all the receivers and since the sender does not keep an explicit list of receivers, termination of RMTP session is timer based [7]. In fact, after the transmission of the last packet, the sender starts a timer that expires after T_{dally} seconds. This interval T_{dally} is set to be twice the lifetime of a packet in an internet. When the timer expires, the sender deletes all state information associated to the connection. Any ACK message from a receiver resets the timer to its initial value. Once the receiver gets all the packets, it stops sending ACK messages.

The main disadvantage of this protocol is that the Sender and DRs have an additional cache. In fact, as discussed above, RMTP provides a high degree of scalability

because the sender does not keep a track of the receivers in the multicast tree. But in order to be able to retransmit the lost packets, the Sender and the DRs have to buffer the message sent during the session and need then an additional cache. Another disadvantage discussed in [7] is that the DRs are chosen by the receivers based on the TTL of the SEND_ACK_TOME packet; and a big number of receivers can choose the same DR, which result in an unbalanced tree.

3) Multicast File Transfer Protocol (MFTP): MFTP [79], which is developed by StarBurst Communication, runs over UDP in the application layer and offers a reliable transfer of files from one sender to multiple receivers. MFTP source switches between three modes: unicast, multicast or broadcast depending on the type of data transmitted and the transmission mode supported by the network [40]. This protocol has two components:

- ***i.*** The *administrative protocol*: used for the group management and for the creation and rupture of the group.
- ***ii.*** The *data transfer protocol*: used for the transmission of the file reliably to all the users in the multicast group.

Using the second component, the MFTP server informs (announcement) a file transfer session and clients join thus the session to receive the file transmitted. The file is sent in passes: First, the sender sends the entire file in the first pass, and retransmits then lost packets in the succeeding passes [40]. When a receiver detects a lost packet, it sends a NACK message to the sender, which retransmits the missing packet. Receivers have to acknowledge reception of the packets only when they are asked to. Announcements are

sent to a *public group* supported by the administrative protocol. However, data is sent on a private group address [40]. Clients can join the public group and listen to the announcements, however, only those allowed by the source can join the private group.

The MFTP server is responsible for the multicast group management, file transfer, and transfer operation. When requested only, receivers acknowledge the reception of the file transmitted.

4) A Tree-Based Reliable Multicast Protocol (TRAM): TRAM [80] is a scalable transport protocol implemented to be used for the transfer of bulk data from a single sender to multiple receivers [59]. This protocol has been a part of the JAVA Reliable Multicast Service JRMS project [59]. Similarly to RMTP [7] and TMTP [46], TRAM is a tree-based protocol, where the tree design is dynamic as in TMTP [46]. TRAM offers different types of trees to satisfy the applications' requirement. The tree is composed of the sender at the root and the receivers at the leaf nodes. Some receivers are designated to be repair heads and are responsible for the repair process of their group members [59]. TRAM is dynamic and allows the redesign of the tree by enabling receivers to find a better head repair.

TRAM is designed in a way to guarantee the data delivery for all of the receivers of the multicast tree [59]. For this end, the reliability is ensured via the use of ACK messages. Upon the reception of data, receivers send ACK messages to their repair head. The latter is able to detect lost messages and to send the missing packets to the required receivers. In order to be able to retransmit data, the head repair has to have a cache memory that contains the data transmitted [59]. Receivers check the reach-ability periodically. If there

is no answer from one receiver, it is then deleted from the repair tree. Similarly, if there is no answer from a head repair, then it is abandoned and its group members are assigned to an active head repair [59]. Depending on the network congestion, the sender transmits data at a variable rate in order to control the flow.

3.4. Summary

Multicasting refers to the delivery of information to a large number of destinations simultaneously. Multicasting has been the first choice of several communication applications because it requires less bandwidth and delays than a multiple unicasting. To that end, many multicast transport protocols have been designed. However not all of them were able to satisfy the different application's requirement and they had to have their own features in order meet these requirement. Among the features that the multicast protocol needed to have, we notice: the reliability, congestion and flow control, ordering, error recovery, and group management.

As the number of applications using multicasting has been growing, the number of multicast transport protocols has been growing as well. The latter can be classified into three categories depending on the type of applications:

1) General purpose protocols: RBP [40], MTP [69], RMP [76], and XTP[42,77]

2) Multicast interactive applications: MTP-2 [82], RTP [81], SRM [10], and RAMP[45]

3) Data Distribution Services: TMTP [46], RMTP [7], MFTP [79], TRAM [80].

The next chapter will present our proposed protocol the Hybrid Multicast Transport Protocol and will explain its design and implementation.

Chapter 4

A Hybrid Multicast Transport Protocol: Design and Implementation

4.1. Introduction

This chapter presents our new hybrid multicast transport protocol, which we refer as HMTP, which is a combination of the four protocols presented before: SRM [10], SCTP[8, 9], RMTP [7], and SRTP [1, 2]. Our scheme takes advantage of the best characteristics of each of the four protocols. This chapter is structured as follow: The first section initially describes the design of the Hybrid Multicast Transport Protocol by illustrating its architecture, the technique applied to ensure the reliability and error recovery, as well as the ordering and scalability. The second section presents the implementation of our Hybrid Multicast Transport protocol.

4.2. Design of the Hybrid Multicast Transport Protocol

4.2.1. Architecture of the HMTP

As presented before, several protocols have been designed in order to find an efficient scheme that can satisfy all Collaborative Virtual Environment (CVE) requirements: reliability, minimum delay, scalability, and synchronization. However, every time a protocol is designed, it concentrates only on one aspect and fails to meet the other requirement necessary for the CVE applications. Each of the proposed protocols presented important issues, such as bandwidth for SCTP and SRM, ordering in RMTP, or even synchronization in SRTP. Since the number of applications that use CVE is growing, the design of an efficient protocol has become a necessity. In order to reach this goal, a hybrid solution has been chosen.

The architecture chosen for our proposed scheme is the *Client-Server* architecture. As described in the Chapter 2, a server (main node) is responsible for the virtual environment's management. It stores information about the different objects and elements of the virtual environment, and keeps a list of the clients connected to it. The latter have to take the permission from the server before accessing any virtual object. When they have the permission, the clients can access the virtual object and change its state. Once they finish processing it, they have to send the new object's state to the server, who routs the information to the interested clients by sending them an update message. If two clients or more are accessing two different objects, then when they finish processing, they have to send the new objects' states to the server. The latter computes the new virtual environment's state and sends it to the connected clients. The server is also responsible for the order of the events.

The virtual environment is represented by a multicast tree. Our protocol is based on hierarchical structure in which users that belong to the same region are grouped together, and one receiver or client is selected as a DR, *Designated Receiver*. DR is basically chosen if it has the largest value of Time to Live (TTL) when sending the SEND_ACK_TOME packet. In fact, DRs and the sender have to send periodically to the receivers a special packet called SEND_ACK_TOME where the TTL field is assigned to a specific value. Receivers checks the value of the TTL of each packet received and select the one that has the highest value. The one selected is considered the nearest DR to the receiver.

The proposed protocol supports a multi-level hierarchy of local regions. In this case, the sender (server) receives ACK messages only from the DRs of the highest level. The DRs of a lower level will be the receivers from DRs of a higher level and senders for the receivers of their own local region. This architecture has been chosen to *minimize the number of messages* sent and to avoid the problem of ACK implosion that the other protocols, such as SCTP [8, 9], encountered. It is also used as a *Congestion Control* plan since the server controls the traffic in the network and the rate of sending packets based on the feedback received from the network. This feedback illustrates the delay and the rate of lost packets. Depending on the feedback, the server changes the rate of sending packets. As previously mentioned, RMTP [7] does not have to know how many members a DR has in its local region. This means that there is a good probability that the information will be missed by some receivers: they are going to miss either all or some of the packets, and then ask for a re-transmission. In order to avoid this problem and the unnecessary re-transmissions, we are going to add the following feature: users can join a

local group, but they have to inform the DR or the sender ahead of time by sending an adhesion message. This approach is used by SRM [10]. In this case, we can ensure that the DR has a specific list of all of the receivers and that will thus send the message to all the members in the local region. New users can send requests for all the packets that they have missed. In this case, the DR sends the packets by applying the immediate transmission request approach described in Section 3.3.3.

The server has to send periodically its status in a "*Status message*" to the clients to keep them informed that he is alive. The "*Status message*" has to be sent reliably to the DRs of the highest level. If the clients do not receive the "*Status message*", they send reliably an "*Inquiry-Status message*" to the server inquiring about its status.

In what follows, we will present the message's format in the proposed protocol, which appears in the form of an *Interaction Stream*, i.e., a sequence of update messages related to a sequence of a user's interactions with a shared object is grouped into one stream [9]. This approach is described above for SCTP [8, 9] in Section 2.5.1. The interaction stream is composed of two kinds of messages: key-update messages and update messages. The update messages represent the position or the motion of an object. The key-update message represents the last update message sent, a final state of an object, or an update message that has to be delivered reliably. The packet's header carries information that allows the knowledge of whether the packet is a key-update message. The packet also allows the identification which shared object is involved in an interaction with, the current interaction stream for this object, and the position of the specific update message in the current stream. The notion of Stream Interaction is really important in

CVEs. It allows the users to know which object is being shared at that moment and to have a stream of all the interactions done with that object.

In our work, the messages will be sent depending on their type. This technique is used in SRTP. Our protocol will use one of the above three modes of transport defined by SRTP. The message sent will be either an "Update message" or a "Key-update message". Update messages are for the data that change frequently (e.g. position). They do not need to be transferred reliably and will thus always be transferred using the mode 0. The key-update messages, on the other hand, need to be transferred reliably. For these messages, we will use either the mode 1 or the mode 2. Specifically, if the key-update message has to be sent reliably to a group of members, mode 1 is used. However, if the key-update message has to be delivered to only one member, e.g. in the case of a retransmission of a lost packet, then mode 2 is used.

For mode 0, which involves data that change frequently, best-effort service is required. This kind of data can tolerate errors; in fact, even if a piece of data is missed, the one that is delivered immediately after will replace it. This does not affect the performance of the protocol. In mode 1, for data that must be received reliably by all members that are in the same multicast group, the NACK approach is used to ensure reliability. In mode 2, however, for a transaction with a single, determined receiver in the multicast group, reliable and timely transport is required and this is accomplished by applying TCP.

In the case of the motion of an object, we will add the timer feature. In fact, as the update messages represent the data of an object that is constantly moving, we assume that these messages are sent periodically in specific increments of time. Then if a receiver

does not receive an update message within that period, it sends a NACK message requesting the lost packet. Of course, if the object is no longer moving, then a key-update message should have been sent, and the receiver will not wait for another message but rather keep its timer to 0. In order to ensure the collaboration, synchronization, and minimum delay, an end-to-end delay is limited to no more than 100 msec for update messages. This limit was proposed in [8, 9], and is going to be used in the proposed protocol.

4.2.2. Reliability, Detection and Retransmission of Lost Packets

DIS, CVE, and C-HAVE applications require a reliable transport for some packets in certain cases. A reliable protocol is then obligatory for these applications. The reliability of our proposed protocol is applied by the NACK messages for the mode 1. This approach is used by SRM [10]. The detection of a lost packet is assured by a timer: specifically, the members that are waiting for a message and do not receive it within the specified period of time send a NACK message to their DR that contains the sequence number of the last packet received. Moreover, if the receiver gets an unexpected packet, it puts the received packet in its buffer and sends a NACK message. The use of the NACK-based approach was chosen in order to avoid the ACK implosion problem.

When the DR sends a message to a receiver using mode 2, reliability is ensured via the ACK messages, and the detection of a lost packet is assured by a timer. After sending the message, the DR starts a timer that will be cancelled if an ACK message (sent from the receiver) is received. Otherwise, if the timer times out, the DR retransmits the message.

The retransmission of the lost packet is the task of the Designated Receiver in the multicast tree. When a receiver finds out that it has not received a packet, it sends a NACK to its DR to inform it about the lost packet. The DR receives the NACK message containing the sequence number of the missed packet, and it sends (unicast) the lost packet to the corresponding receiver. If there is more than one receiver, then the DR has to multicast the lost packet to the receivers. In this case, Mode 1 is used in order to ensure the reliability of the message. Since lost packets are recovered by local retransmissions as opposed to retransmissions from the original sender, *end-to-end latency* is significantly reduced, and the *overall throughput* is improved as well.

4.2.3. Ordering and Scalability

Although some of the protocols described above, such as SRM [10], do not satisfy the ordering requirement, the proposed protocol maintains the ordering through the use of sequence numbers. In fact, every packet delivered contains a sequence number that helps the receiver to order the packet received and to detect a lost packet if there is a gap in the sequence numbers. From its side, the server is responsible for ordering the events that occur. When the clients finish the modification of the objects' states, they send the new state to the server. The latter orders the events that occurred, computes the new virtual environment's state and sends it to the connected clients.

In our work, we combine four protocols that ensure the scalability of the targeted applications. The proposed protocol is designed then to be scalable and fast as follow: Every time a client wants to join the multicast group, it has to inform the sender and the DRs by sending an *adhesion message*. When the members receive this message, they

update their information and they send the message SEND_ACK_TOME, so that the new member can elect its Designated Receiver.

4.3. Implementation of the Hybrid Multicast Transport Protocol (HMTP)

The Hybrid Multicast Transport Protocol is designed and implemented to illustrate its efficiency in satisfying all CVE requirements: minimum delay, synchronization, scalability, and reliability. HMTP is based on a hierarchical structure forming a multicast tree with the sender in the root and the receivers in the leaf nodes grouped into local regions based on their proximity in the network.

The message format is in the form of an *Interaction Stream* composed of *key-update messages* and *update messages*. When an update message is received from the application and has to be sent over the network, a sequence number is appended to this update message. This sequence number equals the maximum value of received and sent sequence numbers for this index plus one, and it is related to a sequence of the user's interactions with a shared object. This update message should be changed to a key-update message if it represents a reference in the application, a final state of an object or an update message that has to be sent reliably to one member.

In our protocol, we have three types of messages: a data packet, a NACK message, or an ACK message. The data packet can be either a regular update message or a key update message and contains information about the virtual environment. The NACK message is sent, when a lost packet is detected. And an ACK message is sent to confirm the reception of

a retransmitted message. Figure 4.10 gives the packet format of the different messages used in the Hybrid Multicast Transport Protocol.

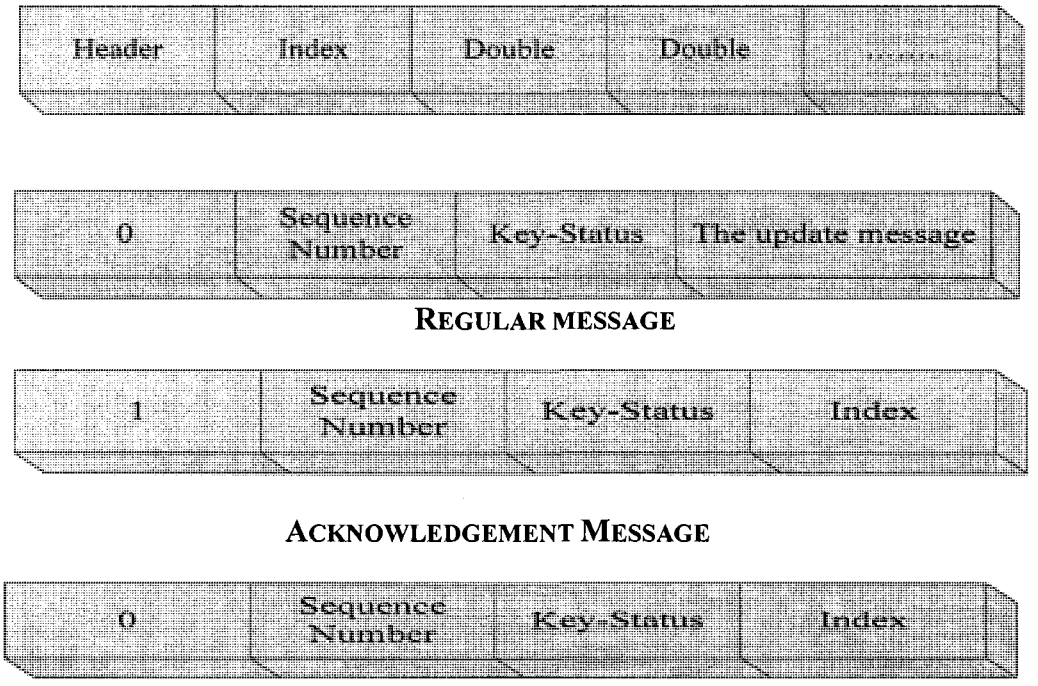


FIGURE 4.10: PACKET FORMAT OF THE MESSAGES

The first field, which is the header, allows the receivers to know the type of the message sent. If the field contains the number 0, then it can be either an update message or a NACK message. However if the first field contains the number 1, then it is an ACK message.

A field *Index* has been added to both ACK and NACK messages. This field is associated with a timer that allows the sender to send the same message again if it does not receive a reply before the timeout. The index is composed of the following information: the sequence number of the message either lost or acknowledged, its index, and the destination ID. Both the ACK and NACK messages do not contain data information such as the update message.

The actions of both the sender and the receiver are described in the pseudo code given below.

Sender Side:

```
1 if update = key update then
2    created timer
3    stored index, sequence number
4    if unicast transmission then
5       stored destination ID
6       //mode 2
7     else
8      number of host collected from session server
9    end if
10   send the update message
11   //mode 1
12   else
13    send the update message with best effort
14   //mode 0
15 end if
```

Receiver Side:

```
1 While(timer is on)
2   Launch a new thread to receive and process on
packet
3   If update is received then
4     If update is ACK then
5      if unicast transmission then
6        cancelled the corresponding timer
7      else
8          decrease the counter or cancel the timer
9        end if
10    else if  update is NACK then
11     send the required update to the designated
nodes reliably
12    //mode two transmission
13   else
14    extract the data from packet
15    //begin with the byte 0 and having data field
16   end if
17   else
18    send NACK to DR
19   end if
20 end while
```

As described in the Section 4.2.2, when a lost packet is detected, the client sends a NACK message to his DR. The latter is responsible for the retransmission of the lost packet. The packet is sent using mode 2 (unicast), if there is only one client, and is sent using the mode 1 otherwise (more than one client). In the last case, the DR keeps a track of the number of clients that did not receive the packet. For both cases, mode 1 and mode 2, the receivers have to send an ACK message to confirm the reception of the lost packet. The DR also starts a timer during the retransmission of the lost packet, and schedules another retransmission of the same message in case of a timeout.

When the DR receives an ACK message, it verifies if the corresponding message was sent to only to one client (via unicast / mode 2) or to several clients (using mode 1). In the first case, unicast, the corresponding timer of the DR, and the retransmission that was scheduled are cancelled. In the case of several clients, the counter is decreased by 1. On the other hand, the receiver launches a new thread periodically to receive and process a packet from the network layer. If it receives an ACK, the corresponding timer is changed based on unicast or multicast transmission. If the packet is NACK (beginning with the byte 0 but no data), it sends the corresponding data reliably. In the case of a data packet (beginning with the byte 0), it extracts the data. When the receiver finds out that it did not receive a packet, by finding a gap in the sequence numbers, it sends a NACK to its DR to inform it about the lost packet, with the index and sequence number of the packet received.

We implemented a Session Manager (SM) that accepts TCP connections by listening to a predetermined port, and maintains a table of users currently connected. The SM provides a set of strings identifying participants that are already members of the session to new participants. SM ensures the scalability of the protocol and also handles the security for the

session. In fact, when a new client wants to join a session, it sends an adhesion message that contains its IP address and its port to the server. The latter transfers the message to the Session Manager, which authenticates and identifies the new user and decides then whether it can join the session or not. If the user is accepted, it is then added to the local hosts table. The SM informs the server about its decision and the new user is then assigned to a DR that also keeps a track of its address. If the user wants to quit the session, it has to inform the server first by sending a message. The server informs the Session Manager, which withdraws the IP address and port number of the user from the session list. The SM confirms then that the user is withdrawn form the list so the information can be updated.

4.4. Summary

In this chapter, we have discussed the features of our proposed protocol, which we refer to as HMTP. Our scheme is based upon a combination of four protocols discussed earlier (i.e., SRTP [1, 2], RMTP [7], SCTP [8, 9] and SRM [10]). Our hybrid Multicast Transport Protocol (HMTP) takes advantage of these protocols and satisfies the CVE requirement, i.e., scalability, minimum delay, reliability and synchronization.

The HMTP is based on the client-server architecture, where the users that belong to the same region are grouped together and where the server is responsible for the management of the virtual environment. The latter is represented by a multicast tree. The message is in the form of an interaction stream composed of update and key-update messages. Update messages are for data that change frequently and do not need to be sent reliably and will be thus transferred using Mode 0: best effort. Key-update messages represent a reference and need then to be transferred reliably. Mode 1 and 2 are used for

this end. Mode 1 is used if the message has to be sent reliably for a group of users; and Mode 2 is used to send messages reliably to a single determined user. The NACK approach is used to ensure the reliability and if a message is lost, than it is recovered by local retransmission. This reduces the end-to-end latency and improves the overall throughput.

Chapter 5

Implementation of the Hybrid Multicast Transport Protocol (HMTP) in Virtual Tele-Surgery Training Applications

5.1. Introduction

In this chapter, we will describe the implementation of our proposed protocol, Hybrid Multicast Transport Protocol (HMTP), which is designed for Collaborative Virtual Environments (CVE) applications that use haptic devices. We will also discuss the two applications we have used to evaluate the performance of our protocol, i.e., the tracheotomy tele-surgery and brain tumor tele-Surgery application.

5.2. Architecture of the Tele-Surgery Application

The Hybrid Multicast Transport Protocol can be used in many applications based on Collaborative Virtual Environments and using haptic devices. The Figure 5.11 illustrates the architecture of the Tele-Surgery applications designed, and developed at the PARADISE and DISCOVER research laboratories at the University of Ottawa. The different components of the application are described in details below.

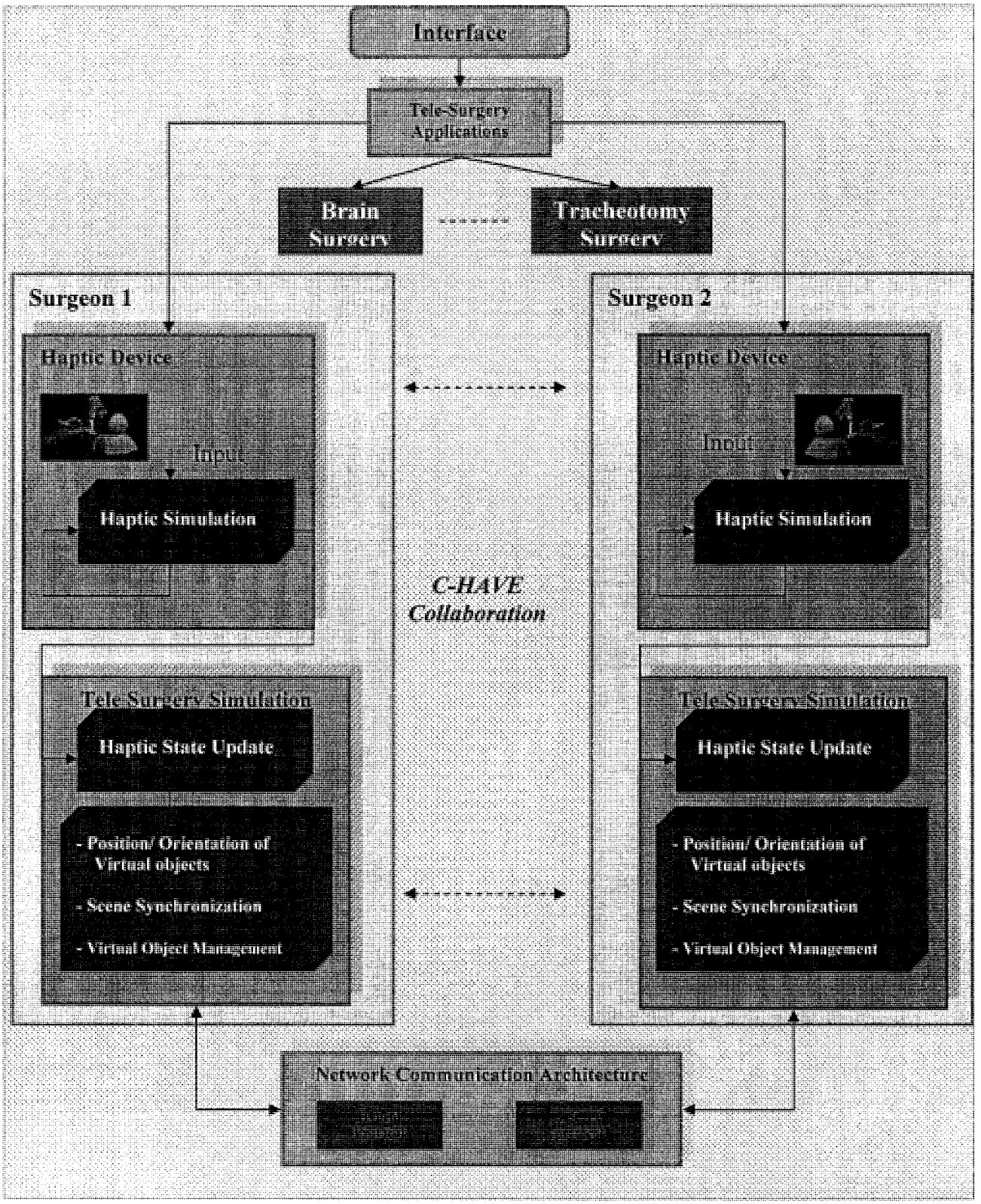


FIGURE 5.11: MAIN ARCHITECTURE FOR TELE-SURGERY APPLICATIONS

As shown in Figure 5.11 the same architecture is applied for both Tele-Surgery applications: the surgeon chooses which application he wants to run and which Network Communication protocol he wants to use. Once the specifications have been chosen, the two surgeons are able to collaborate together using haptic devices and manipulate the same virtual object. The Tele-Surgery application's architecture has four main modules:

1. The Interface module
2. The Tele-Surgery Application module
3. The Communication Architecture module
4. The User module

1) The Interface Module: shown in Figure 5.12, is actually composed of a GUI that allows the surgeons to choose a specific view of the patient. In our applications, we designed different views such as: the top view of the patient and the general view of the operating room.

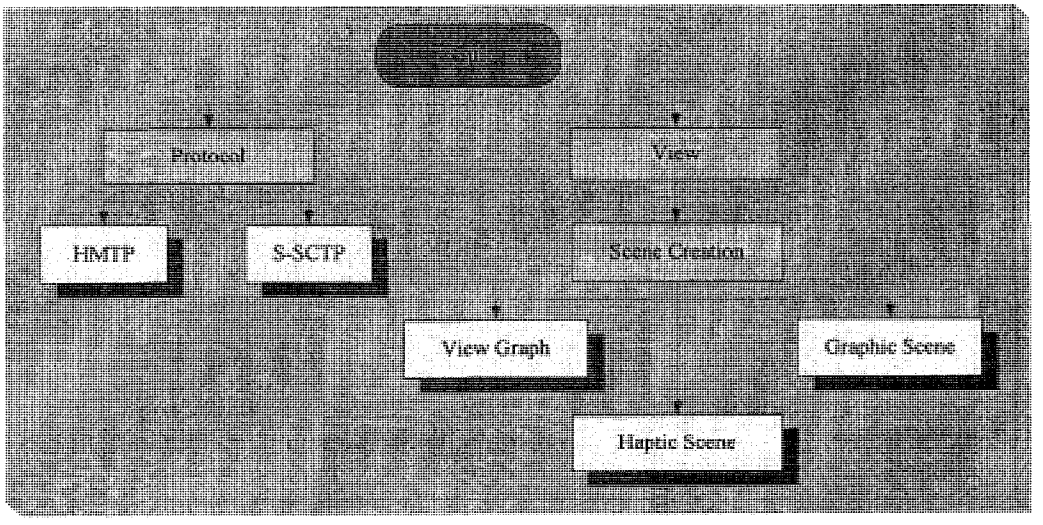


FIGURE 5.12. THE INTERFACE MODULE

Once the view is chosen, the virtual scene is created, the haptic device, PHANToM [61][2], is reset and the Tele-Surgery Application is run. The virtual scene is composed of the graphic scene, the view graph and the haptic scene. When the scene is created, the surgeons can then choose the protocol they want to use. This feature was added in order to show the efficiency of the hybrid multicast transport protocol compared to the other protocols such as S-SCTP [18, 21].

2) The Tele-Surgery Application Module: it illustrates the different applications on which we are working. Two Tele-surgery applications have been carried out in the DISCOVER Research Laboratory at the University of Ottawa: the Tracheotomy Tele-Surgery application, and the Brain Tumor Tele-Surgery application. Both applications are described in details below. These applications have training purposes and allow surgeons to manipulate a shared virtual object via the use of haptic devices such as the PHANToM [61]. The latter, as described in Section 2.3.2, offers a General Haptics Open Software Toolkit (GHOST ® SDK) [70], which is a C++ software toolkit that helps developing touch-enabled applications. The Tele-Surgery applications developed were written in Java, the Java Native Interface (JNI) which has been used to communicate with the C++ program using General Haptic Open Software Toolkit (GHOST® SDK) [70]. The JNI provides "native" methods to call the Microsoft Dynamic Loadable Library (DLL) from the Java program. The description of a Java3D universe is the description of a simple universe, used in our implementation to provide a complex viewing architecture, separated from the scene graph. VRML descriptions are loaded in Java3D and handle avatar behaviors.

[2] The Personal Haptic Interface Mechanism

3) The Communication Architecture Module: it represents the transport protocol used during the Tele-surgery application. This protocol has to ensure the collaboration and synchronization between the surgeons. For both Tele-surgery applications developed, we are using the Hybrid multicast transport protocol and the S-SCTP [18, 21], the latter of which was chosen in order to show the efficiency of our hybrid multicast transport protocol.

4) The User Module: it manipulates two sub-modules: the haptic device and the tele-surgery simulation modules. When the surgeon chooses the Tele-surgery application, a simulation of that application is run on his station.

In both Tele-surgery applications developed, the surgeons are using the PHANToM Omni device [61], which provides force feedback between the virtual object and the user. Every time one surgeon moves the PHANToM [61], the haptic device does an internal simulation in order to calculate the force feedback and to determine the modifications that have to be done on the shared object. This output is sent to the Tele-surgery simulation module in order to update the shared object's state. Once the modifications are applied, they are immediately sent via one of the protocols (Hybrid Multicast Transport Protocol or S-SCTP [18, 21]) in order to update the shared object's state on the second surgeon's Tele-surgery simulation. By applying this technique, synchronization is maintained between the surgeons.

Tele-Surgery applications require that the users collaborate in closely coupled and highly synchronized tasks to manipulate a shared object, the patient. By adding haptic devices to the application, the degree of synchronization becomes higher and the use of an efficient protocol, HMTP that ensures a good communication is important. The next sections are devoted to giving a detailed description of each Tele-Surgery application carried out

jointly at the PARADISE and DISCOVER research laboratories.

5.3. The Haptic Virtual Reality Tracheotomy Tele-Surgery Application

Tracheotomy Tele-Surgery is an application that aims to simulate a surgical act performed in emergency medicine. This surgery requires very tightly coupled collaboration between members of the surgery team. In fact, a lack of collaboration can lead to fatal errors. This Tele-surgery application has training purposes and aims to allow the surgeons to have the required training in order to perform the surgery successfully. In this scenario, we presented a simple Tele-surgery application in which two surgeons, or trainees, must share tools such as a scalpel, surgical hooks, and a piece of gauze, to cut the skin of a virtual patient's throat, spread it open, and then cut inside the underling muscle layer.

The Haptic Interface is implemented by JNI and a C++ DLL to the GHOST SDK [70]. It loads a VRML description of the throat in the haptic interface and allows a method that returns the PHANToM [61] position and orientation. The following Figure 5.13 gives a snapshot of the top view of the patient.

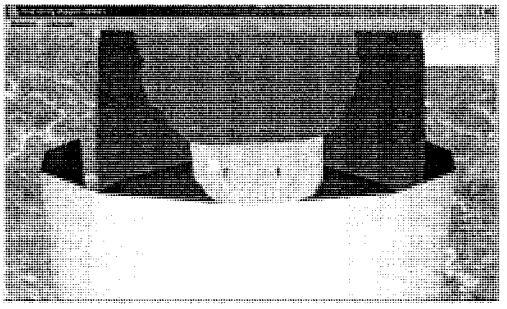
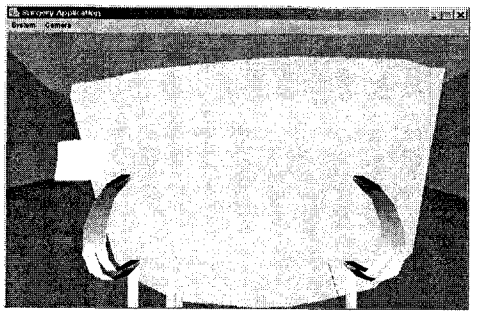
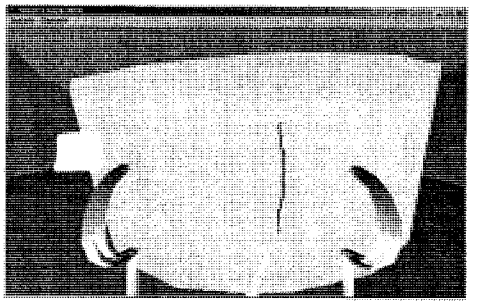


FIGURE 5.13: TOP VIEW OF THE PATIENT IN THE TRACHEOTOMY TELE-SURGERY

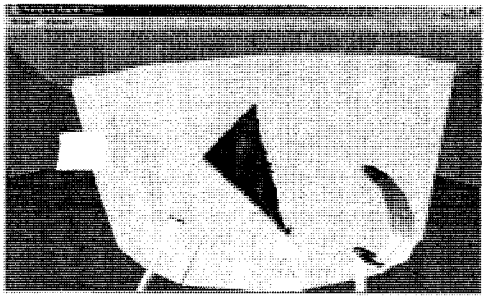
During each simulation session, one surgeon takes the scalpel and performs a vertical cut while the other surgeon uses the gauze to remove spilled blood. The first surgeon successively takes both hooks to open the skin, while the second surgeon performs a horizontal cut of the muscle. These actions constitute a successful operation and each is illustrated in Figure 5.14 below.



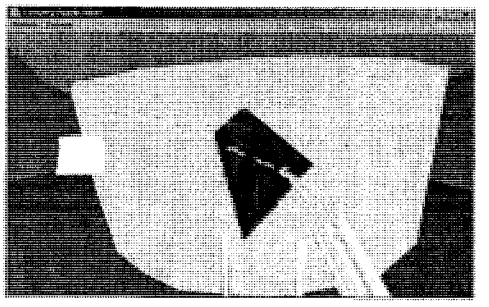
a) Operation area



b) Performs vertical cut.



c) Pull the skin.



d) Cut on the muscle

FIGURE 5.14: SUCCESSFUL COLLABORATION IN THE TRACHEOTOMY TELE-SURGERY APPLICATION.

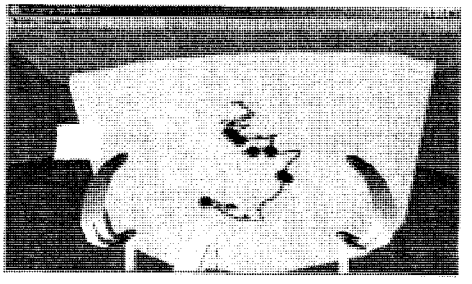
The above Tele-surgery application was tested between two users, one acting as the first surgeon and the other acting as the second. Their respective duties were explained. The success of the surgery depended on close collaboration between the remote surgeons.

Collaboration's ***failure*** can always happen and is expressed in two ways:

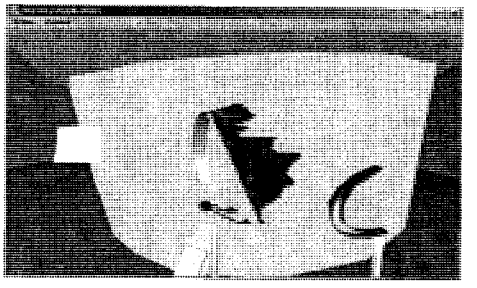
i. One of the surgeons cuts the wrong place because he is not trained enough and does not know where he is supposed to cut. This failure is expressed in a big amount of bleeding and the skin being unable to be pulled.

ii. One of the surgeons cuts the wrong place of the muscle resulting in a big amount of bleeding.

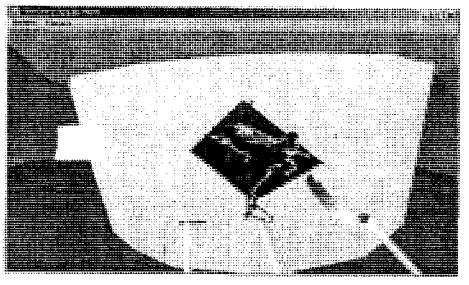
The collaboration's failure occurs because of network problems such as the network delay, the jitter and the loss of update messages. Figure 5.15 illustrates an unsuccessful surgery where the failures describes above occur.



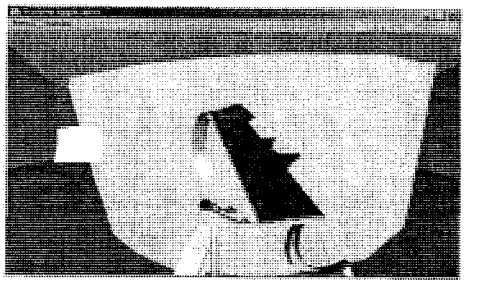
a) Jagged (zig-zag) cut.



b) Hook can't pull the skin



c) Cut in wrong place



d) Jagged (zig-zag) cut the muscle.

FIGURE 5.15: FAILURE OF COLLABORATION IN THE TRACHEOTOMY TELE-SURGERY APPLICATION.

In order to show that the collaboration's failure is due to network problems, the application was first tested with no packet loss. In this case, and after performing fifty trials, no failure

was observed. This is a necessary test to ensure that failures, which do occur, are not due to the nature of the application but instead to network delay, jitter, and packet loss.

The next section introduces and presents the Haptic Virtual Reality Brain Tumor Tele-Surgery application.

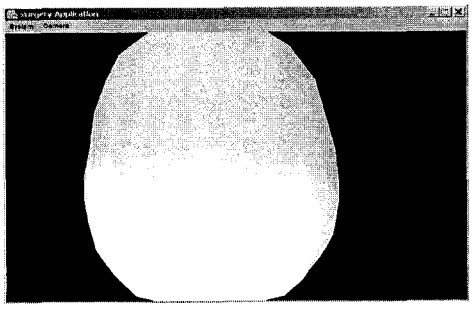
5.4. The Haptic Virtual Reality Brain Tumor Tele-Surgery Application

The Brain Tumor Tele-surgery application has training purposes and aims to instruct the surgeons on how to perform the Brain tumor Tele-surgery effectively. Haptic interfaces that have been designed for brain surgery simulations may prove to be especially useful for training surgeons to conduct minimally invasive procedures and remote surgery using Tele-operators. We designed a virtual patient's head by representing his actual skin, skull bone, brain, and tumor. Haptic interfaces provide soft feelings for the skin and brain as well as a hard feeling for the skull bone. In order to perform the surgery and remove the tumor, the users start by downloading the patient's body (head) and collaborate together.

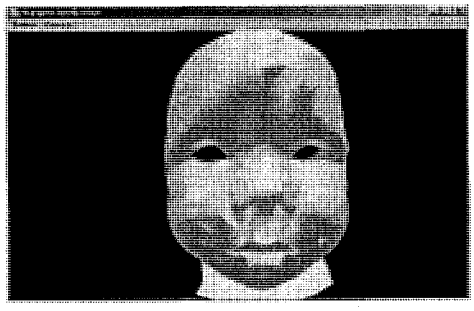
The procedure involves shaving a portion of the head and making an incision in the scalp, then using medical tools to remove a portion of the skull. This enables the participants to find and remove the tumor. After the tumor is removed, the portion of the cut skull is replaced. Brain mapping; during the surgery simulation; gives the trainee valuable information about the brain tumor. Both the expert neurosurgeon and trainee precisely identify sensitive areas of the brain that are responsible for speech, comprehension, sensation or movement.

The Brain Tumor Tele-Surgery is an application that aims to simulate a surgical act

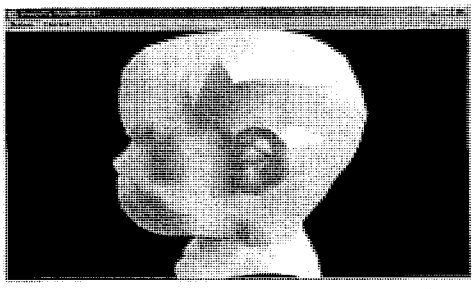
commonly performed on a critical part of the human body. This surgery requires very tightly coupled collaboration between participants. In this scenario, we presented a simple Tele-surgery application where two surgeons, or trainees, must share tools such as a scalpel, surgical hooks, and a piece of gauze, in order to cut the scalp of a virtual patient's head, and remove part of the skull to see the brain and tumor inside. Snapshots of the three-dimensional human head used for the brain tumor tele-surgery application are shown in Figure 5.16.



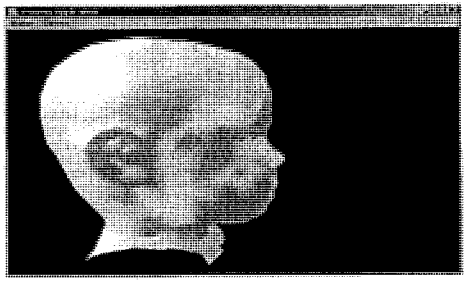
a) Human Head



b) Operation area



c) Left side

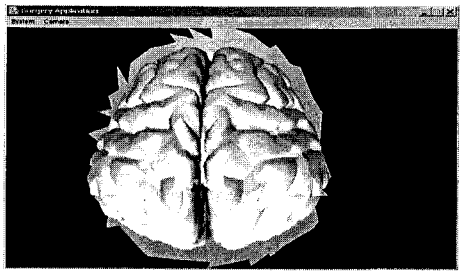


d) Right Side

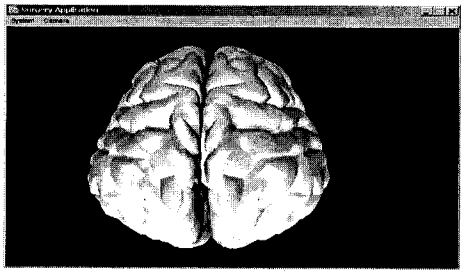
FIGURE 5.16: THREE-DIMENSIONAL HUMAN HEAD BEFORE THE BRAIN-TUMOR TELE-SURGERY.

During each simulation session, one surgeon uses the scalpel to perform a vertical cut while the other surgeon uses the gauze to remove the spilled blood (we use a square to denote the skin being cut). The first surgeon successfully cuts away a section of the skull, while the second surgeon pulls the skull piece away from the head. Once it has been removed, the brain

and interior of the skull will be easily visible from the top and sides. For learning purposes, we can even take out the brain. Screenshots of the brain are illustrated in Figure 5.17.



a) Top view



b) Brain part

FIGURE 5.17. SCREENSHOTS OF THE BRAIN.

The same type of failures, described in the Section 5.3, can occur for the brain-tumor tele-surgery application. The failures are expressed by a big amount of bleeding when one of the surgeons cut the wrong place of the skin. The failures occur because of network problems, and in order to show that, the application was tested fifty times with no packet loss. In all the cases, there was no failure.

5.5 Summary

In this chapter, we described the two tele-surgery applications that we applied our hybrid multicast protocol on: the tracheotomy tele-surgery application and the Brain tumor tele-surgery application. We also described the failures that can occur in both applications.

The next Chapter 6 is devoted to illustrate the performance of the Hybrid Multicast Transport Protocol and to give the experimental results obtained when applied in both tele-surgery applications.

Chapter 6

Experimental Results

6.1. Introduction

In this chapter, we present our experimental results and evaluate the performance of our proposed protocol. As mentioned before, the Hybrid Multicast Transport Protocol was implemented and applied to satisfy CVE requirement i.e. scalability, reliability, minimum delay and synchronization. In order to show it, the HMTP was applied to both Tele-surgery applications developed in the PARADISE and DISCOVER research laboratories: the Tracheotomy and Brain-tumor Tele-surgeries.

This chapter is structured as follow: the first section illustrates the results obtained when applying our proposed protocol to the Tracheotomy Tele-surgery application, while the second section gives the results gotten when using our HMTP for the Brain Tumor Tele-surgery application.

6.2. Results of the Haptic Virtual Reality Tracheotomy Tele-Surgery Application

As mentioned in Chapter 5, the haptic virtual reality tracheotomy Tele-surgery application was tested first with no network problems. This test was performed in order to show that the collaboration failures do occur and are caused by network problems. These tests were then performed in scenarios in which there was no lag, as well as scenarios in which there were loss rates of 20%, 30% and 50%. We performed the same tests again by adding a 50ms delay. Network delay was simulated by an intermediate node acting as a router that would delay or drop packets. These tests were done using two protocols: S-SCTP [18, 21] and Hybrid Multicast Transport Protocol HMTP. As mentioned in Section 2.4.1, Smoothed SCTP (S-SCTP) [18, 21] was developed in order to overcome the jitter problem that SCTP [8, 9] encountered. S-SCTP is thus an enhanced version of SCTP, and we judged it better to use S-SCTP than SCTP in our experiments to illustrate the efficiency of HMTP.

Table 1. Results of the Tele-Surgery Application Based on the Number of Failures.

Network conditions	S-SCTP		HMTP	
	NOT (Number of Trails)	NOF (Number of Failures)	NOT (Number of Trails)	NOF (Number of Failures)
No lag	50	9	50	9
20% loss	50	13	50	9
30% loss	50	14	50	10
50% loss	50	18	50	11
50ms delay	50	17	50	13

A group of four participants was used to conduct this experiment. For consideration of execution time, there were five trials, each involving normal reverse order experiments to avoid the "training effect". The participants had to execute the required actions described in Section 5.2 in order to perform the tele-surgery. For communication, S-SCTP [18, 21] and Hybrid Multicast Transport Protocol (HMTP) were used. Collaboration failures were defined in Section 5.2. Fifty trials were processed using S-SCTP and HMTP to test the Tele-surgery application. Table 1 and Figure 6.18 show the test results for the failures that occurred when using these two protocols and varying the network conditions. The results show that the occurrence of failures is reduced in the Hybrid Multicast Transport Protocol-based Tele-surgery application, as opposed to scenarios using the S-SCTP based Tele-surgery model.

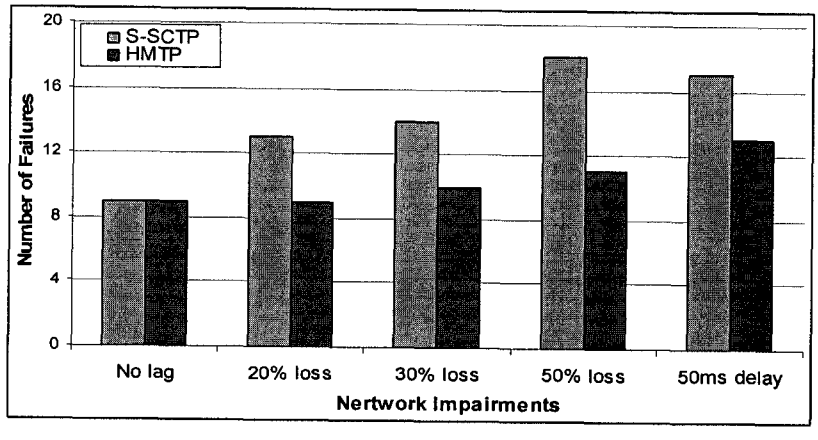


FIGURE 6.18. DEPENDENCY BETWEEN NUMBER OF FAILURES AND NETWORK IMPAIRMENTS.

Table 2. Results of the Tele-Surgery Application Based on the Execution Time Needed to Complete the Application

Network conditions	S-SCTP	HMTP
No lag	180.3	166.8
20% loss	200.5	175.5
30% loss	205.0	176.8
50% loss	196.0	170.0
50ms delay	212.8	171.0

A second type of tests was done in order to illustrate which protocol has the smallest total execution time. The total execution time is defined in our Tracheotomy application as the difference between the completion times (i.e. when the gauze finishes cleaning the blood spilled after the second cut), and the beginning times (i.e. the time at which the scalpel or the gauze starts moving). All the times are considered in seconds. The results are shown in Table 2 and Figure 6.19. As illustrated, the execution time is reduced using the Hybrid Multicast Transport Protocol in comparison with S-SCTP.

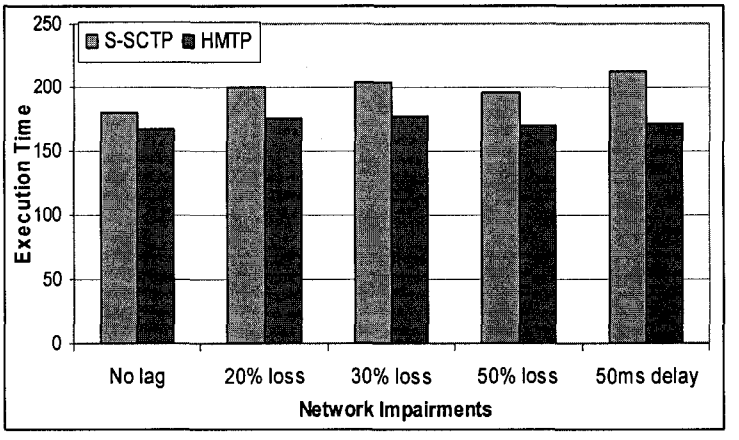


FIGURE 6.19. DEPENDENCY BETWEEN EXECUTION TIME AND NETWORK IMPAIRMENTS.

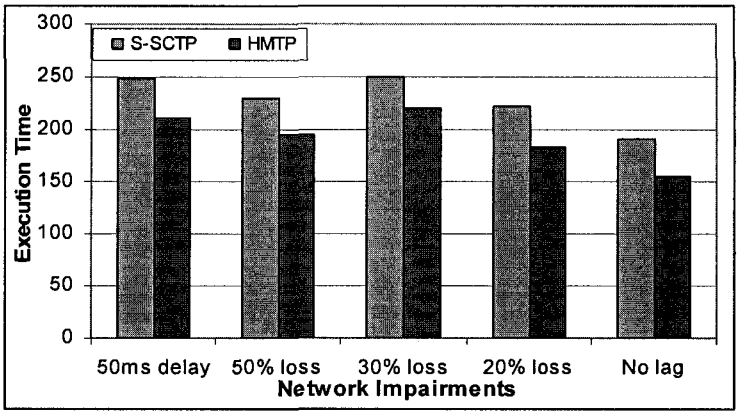


FIGURE 6.20. RESULTS OF THE TELE-SURGERY APPLICATION BASED ON THE NUMBER OF FAILURE DURING TRIALS (REVERSE).

Another type of tests was done in order to show which protocol has the least number of failures, by changing the network conditions and starting from the worst case (50ms delay) and going to the best condition (which is no lag). We know that this type of test is important to show the efficiency of our protocol. In fact, the results may differ if we start from a state of Failure instead of a state of Success. The results of this type of test are shown in Table 3 and Figure 6.20. The results of this type of tests show that the HMTP has the least number of failures as opposed to S-SCTP when applied in the Tele-surgery model and starting from the worst network condition.

Table 3. Results of the Tele-Surgery Application Based on the Execution Time Required to complete the Application (Reverse Order).

Network conditions	S-SCTP	HMTP
50ms delay	248.8	210.5
50% loss	230.4	196.0
30% loss	250.4	220.8
20% loss	222.0	183.0
No lag	191.4	155.0

6.3. Results of the Haptic Virtual Reality Brain Tumor Tele-Surgery Application

As mentioned in Section 5.3, the haptic virtual reality brain tumor Tele-surgery application was first tested with no network problems. No collaboration's failures occurred. In our distributed application, we consider the issues of network impairment such as network delay, jitter, and packet loss. To mitigate the network's drawbacks, we develop and implement our Hybrid Multicast Transport Protocol (HMTP) to allow collaboration in the Brain Tumor Tele-Surgery application. We also implemented the Tele-surgery application based on S-SCTP [18, 21] in order to illustrate the efficiency of HMTP. Network delay was simulated by an intermediate node acting as a router that would delay or drop packets. The tests were performed in scenarios in which there was no lag, as well as scenarios in which there were loss rates of 20%, 30%, and 50%. We performed the same tests again by adding 50ms delay. These tests were done using two protocols: S-SCTP [18, 21] and HMTP.

The same group of participants from the Tracheotomy Tele-Surgery was used to conduct this experiment. 30 trials were processed using S-SCTP and HMTP to test the Brain Tumor Tele-Surgery application. Table 4 and Figure 6.21 show the test results for the failures that occurred when using these two protocols. By varying the network conditions, results show that the occurrence of failures is reduced in the Hybrid Multicast Transport Protocol-based Tele-surgery application as opposed to scenarios using the S-SCTP based model.

Table 4. Results of the Brain Tumor Tele-Surgery Application Based on the Number of Failures

Network conditions	Smooth SCTP		HMTP	
	NOT (Number of Trails)	NOF (Number of Trails)	NOT (Number of Trails)	NOF (Number of Trails)
No lag	30	5	30	2
20% loss	30	7	30	3
30% loss	30	6	30	5
50% loss	30	13	30	8
50ms delay	30	10	30	8

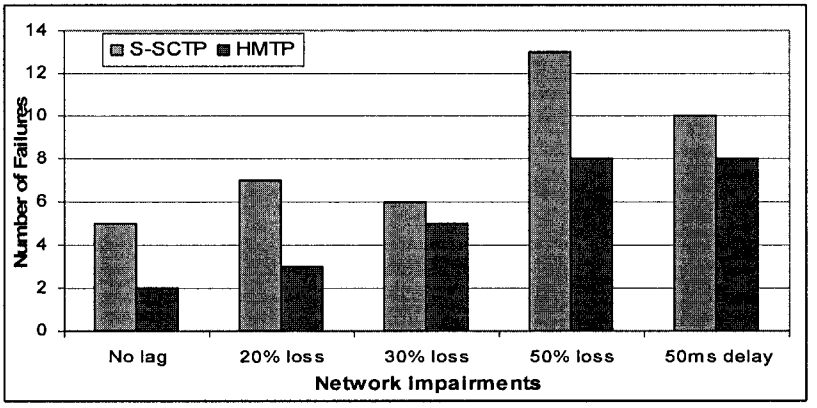


FIGURE 6.21. DEPENDENCY BETWEEN THE NUMBER OF FAILURES AND NETWORK IMPAIRMENTS.

6.4. Summary

In this chapter, we presented the results we have obtained to evaluate the performance of our scheme for two Tele-surgery applications, i.e., Brain Tumor tele-surgery application and the Tracheotomy tele-surgery application.

We used S-SCTP [18, 21], an enhanced version of SCTP [8, 9], in order to show the

efficiency of our proposed protocol. Different tests were performed: variation of network conditions, and dependency between execution time and network impairments. These tests were also performed in the reverse order. As illustrated above, the use of our Hybrid Multicast Transport Protocol in both Tele-surgery applications gives better results than the use of the other transport protocol S-SCTP.

Chapter 7

Conclusion and Future Work

Collaborative Virtual Environment (CVE) and Collaborative Haptic, Audio and Visual Environment (C-HAVE) are used in many scenarios such as Tele-surgery class of applications. Many techniques have been discussed in literature regarding how to satisfy the following CVE requirements: scalability, reliability, synchronization, and minimum delay. However, these protocols have failed to achieve this goal. In order to deal with these issues, in this thesis we have designed an efficient hybrid multicast transport protocol for CVE applications.

7.1. Thesis Contributions and Future Work

In this thesis, we have proposed and designed a hybrid multicasting transport protocol that has the following features:

- ✓ It takes advantage of the best characteristics of the SRTP [1, 2], SRM [10], RMTP [7], and SCTP [8, 9] protocols.
- ✓ It is based on the client server architecture,
- ✓ It uses a multicast tree when transmitting messages, which allows it to avoid the congestion and end-to-end delay problems.
- ✓ It also uses the transport modes described by SRTP in order to ensure reliability.

- ✓ As for the synchronization, our hybrid solution uses the Interaction Stream described in SCTP.

The Hybrid Multicast Transport Protocol (HMTP) was developed, implemented and tested in two different virtual tele-surgery training applications carried out at the DISCOVER and PARADISE research laboratories: the Tracheotomy Tele-Surgery application, and the Brain Tumor Tele-Surgery application. In order to show its performance, the HMTP was compared with the Smoothed SCTP [21], which is an enhanced version of SCTP [8, 9]. Two types of tests were executed: the occurrence of failures and the execution time when changing the network impairments. The obtained results of the two tests indicate that the occurrence of failures and the execution time are reduced when using HMTP as opposed to Smoothed SCTP.

In the future, we plan to investigate further the performance of our scheme and extend it by adding fault tolerant mechanisms. We also plan to extend our protocol towards CVE and C-HAVE applications in a wireless environment using wireless TCP. To do so, we need first to identify haptic devices that have the wireless capabilities.

References

[1] J. M. Pullen and V. P. Laviano, "Adding Congestion Control to the Selectively Reliable Transmission Protocol for Large-Scale Distributed Simulation", *Fall 1997 Simulation Interoperability Workshop paper 97F-SIW-018*, September 1997.

[2] J. M. Pullen and N. Kakarlamudi, "Performance Issues for the Light-Weight RTI", *Proceedings of the 3rd Simulation Interoperability Workshop*, Orlando, FL, September 1998.

[3] J. M. Pullen and V. P. Laviano, "A Selectively Reliable Transport Protocol for Distributed interactive Simulation". *Proceedings of the 13th Workshop on Standards for the Interoperability of Distributed Simulations,* pp. 95-102, 1995.

[4] D. M. Moen and J. M. Pullen, "A Performance Measurement Approach for the Selectively Reliable Multicast Protocol for Distributed Simulation", *Proc. of the Fifth IEEE Workshop on Distributed Simulation and Real-Time Applications*, pp.30-34, 2001.

[5] J. M. Pullen, "Reliable Multicast Network Transport for Distributed Virtual Simulation", *Proc. IEEE Workshop on Distributed Interactive Simulations and Real-Time applications (DISRT '99),* pp. 59-66, 1999.

[6] A. Siafa, "Protocol Multipoint Fiable et ordonné pour applications coopératives assynchrones." PhD thesis, Universite De Savoie 1998.

[7] S. Paul, K. K. Sabnani, J.C. Lin and S. Bhattacharyya, "Reliable Multicast Transport Protocol", *IEEE Journal on Selected Areas in Communications*, pp. 407-421, Vol. 15, 1997.

[8] S. Shirmohammadi and N.D. Georganas, "An End-to-End Communication Architecture for Collaborative Virtual Environments", *Computer Networks Journal,* pp. 351-367, Vol. 35, No. 2, February 2001

[9] S. Shirmohammadi and N. D. Georganas, "Collaborating in 3D Virtual Environments: A Synchronous Architecture", *Proc. IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, Knowledge Media Networking Workshop, National Institute of Standards and Technology (NIST),* U.S.A., pp. 35-42, June 2000.

[10] S. Floyd, V. Jacobson, C.G. Liu, S. McCanne, and Lixia Zhang, "A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing", *IEEE/ACM Transactions on Networking*, pp.784-803, Dec 1997.

[11] D. DeLucia and K. Obraczka, "A Multicast Congestion Control Mechanism for Reliable Multicast", *3rd IEEE Symposium on Computers & Communications*, pp.142, Athens, Greece,1998.

[12] S. K. Kasera, J. Kurose and D. Towsle, "Scalable Reliable Multicast Using Multiple Multicast Groups", *Proc. ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pp. 64-74, 1997.

[13] B. N. Levine, S. Paul and J.J. Garcia-Luna-Aceves, "Organizing Multicast Receivers Deterministically by Packet-Loss Correlation", *Multimedia Systems Journal*, pp. 3-14,Vol 9, No 1, 2003.

[14] P. Parnes, "The mStar Environment. Scalable Distributed Teamwork using IP Multicast", *Master Thesis*, September 1997.

[15] J. M. Pullen, "Limitation of Internet Protocol Suite for Distributed Simulation in the Large Multicast Environment", *Simulation Interoperability Workshop*, Florida, March 1997.

[16] J. K. Shapiro, D. Towsley and Jim Kurose, "Optimization based-Congestion Control for Multicast Communication", *Proceedings of the Second International IFIP-TC6 Networking Conference on Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; and Mobile and Wireless Communications*, pp.423-442, Vol. 2345, 2002.

[17] A. Boukerche, S. Shirmohammadi, A. Hossain, "Prediction Based Decorators for Distributed Collaborative Haptic Virtual Environments", *J. Computer Applications in Technology, Special Issue in Collaborative Multimedia Applications in Technology* 2007 (Pending publication)

[18] A. Hossain, "Virtual Reality Simulation Modeling for Tele-Surgery and Tele-Haptic Class of Applications", *Master Thesis*, University of Ottawa, 2005.

[19] S. Shirmohammadi and N. H. Woo, "Shared Object Manipulation with Decorators in Virtual Environments", *Proc. of the 8thIEEE International Symposium on Distributed Simulation and Real-Time Applications*, pp. 230-233, 2004.

[20] X. Shen, J. Zhou, A. El Saddik, and N.D. Georganas, "Architecture and Evaluation of Tele-Haptic Environments", *Proc. of the Eighth IEEE International Symposium on Distributed Simulation and Real-Time Applications*, pp. 53-60, 2004.

[21] S. Dodeller and N.D. Georganas, "Transport Layer Protocols for Telehaptics Update Messages", *Proc. 22nd Biennial Symposium on Communications*, Queen's Univeristy, Canada, May 31-June3,2004

[23] C. Bouras, E. Giannaka, A. Panagopoulos, T. Tsiatsos "A platform for Virtual Collaborative spaces and educational communities: the case of EVE". *Multimedia Systems Journal, Special Issue on Multimedia System Technologies for Educational Tools, Springer Verlang*, Vol. 11, No. 3, pp. 290 – 303, 2006.

[24] Elizabeth F. Churchill "Collaborative Virtual environments", *Springer; 1 edition* (April 20, 2001), ISBN-10: 1852332441

[25] Encyclopedia of Virtual Environments [http://www.hitl.washington.edu/scivw/EVE/] *Last accessed January 2008.*

[26] Logitech iFeel Mouse [http://www.firingsquad.com/hardware/ifeelmm/default.asp] *Last accessed January 2008.*

[27] Logitech iFeel MouseMan [http://www.deafgamers.com/ifeelmm.htm] *Last accessed January 2008.*

[28] SensAble Technologies [http://www.sensable.com/products-haptic-devices.htm] *Last accessed January 2008.*

[29] CyberGrasp™ Exoskeleton [http://www.immersion.com/3d/products/cyber_grasp.php] *Last accessed January 2008.*

[30] V. Hayward, O. R. Astley, M. Cruz-Hernandez, D. Grant, G. Robles-De-La-Torre, "Haptic Interfaces and Devices". *Sensor Review,* pp. 16–29, Vol. 24, Number 1, Emerald Group Publishing Limited, 2004 ·.

[31] Immersion's Haptic Workstation [http://immr.client.shareholder.com/ReleaseDetail.cfm?ReleaseID=134529] *Last accessed January 2008.*

[32] X. Shen, F. Bogsanyi, L. Ni, and N. D. Georganas."A heterogeneous scalable architecture for collaborative haptics environments". *IEEE International Workshop on Haptic Audio and Visual Environments (HAVE),* pp. 113–118, September 2003.

[33] M. Eid, M. Orozco and A. El Saddik, "A Guided Tour in Haptic Audio Visual Environment and Applications", *International Journal of Advanced Media and Communication*, pp. 265 - 297, Vol.1, N.3, 2007.

[34] C. Gutwin, S. Benford, J. Dyck, M. Fraser, et al., "Revealing delay in collaborative environments", *Proc. ACM Conf. on Human Factors in Computing Systems*, pp. 503–510, Vienna, Austria, 2004,

[35] Emmanuel Frécon, "A Survey of CVE Technologies and Systems", *SICS Technical Report*, T2004-03, ISSN 110-3154, 2004.

[36] El-Far, N.R.; Nourian, S.; Jilin Zhou; Hamam, A.; Xiaojun Shen; Georganas, N.D., "A cataract tele-surgery training application in a hapto-visual collaborative environment running over the CANARIE photonic network", *IEEE Workshop on Haptic Audio Visual Environments and their Applications (HAVE),* October 2005.

[37] C. Oliveira, X. Shen, N. Georganas, "Collaborative Virtual Environment for Industrial Training and e-Commerce", *Workshop on Application of Virtual Reality Technologies for Future Telecommunication Systems, IEEE Globecom*, pp. 288 – 292, 2000.

[38] F. Santos, B. Fonseca, P. Martins, L. Morgado, "Negotiation of Spatial Configurations in Collaborative Virtual Environments", *m-ICTE 2006*, Sevilla, Spain, FORMATEX, 11/2006.

[39] M. Mauve, V. Hilt, C. Kuhmünch, W. Effelsberg, "RTP/I - Towards a Common Application Level Protocol for Distributed Interactive Media". *IEEE Transactions on Multimedia*, pp. 152 – 161, Vol. 3 No. 1, 2001,

[40] K. Obraczka, "Multicast Transport Mechanisms: A Survey and Taxonomy," *IEEE Communications Magazine*, pp. 94-102, Vol. 36, No. 1, 1998.

[41] Multicast Transport Protocol [ftp://ftp.rfc-editor.org/in-notes/rfc1301.txt] *Last accessed January 2008.*

[42] J.W Atwood, "Classification of Reliable Multicast Protocols," *IEEE Network*, pp.24-34, Vol.18, No.3, 2004.

[43] Real Time Protocol [http://icapeople.epfl.ch/thiran/CoursED/RTP.pdf] *Last accessed January 2008.*

[44] RTP: A Transport Protocol for Real-Time Applications [http://www.ietf.org/rfc/rfc1889.txt] *Last accessed January 2008.*

[45] Alex Koifman, Stephen Zabele, "RAMP: a Reliable Adaptive Multicast Protocol", *IEEE Proceedings Fifteenth Annual Joint Conference of the IEEE Computer Societies (INFOCOM'96)*, San Francisco, CA, pp. 1442-1451, March 1996.

[46] R. Yavatkar, J. Griffioen, and M. Sudan, "A Reliable Dissemination Protocol for Interactive Collaborative Applications.", In *Proceedings of the ACM Multimedia '95 Conference*, pp. 333-344, November 1995.

[47] StarBurst Multicast File Transfer Protocol (MFTP) Specification [http://tools.ietf.org/html/draft-miller-mftp-spec-02] *Last accessed January 2008.*

[48] W. Lam, "Multicast Data Dissemination" *PhD Thesis,* 2004.

[49] Towsley, D. Kurose, J. Pingali, S., "A comparison of sender-initiated and receiver-initiated reliablemulticast protocols", *IEEE Journal on Selected Areas in Communications,* pp. 398-406, Vol. 15, Issue 3 , 1997.

[50] Zhen Xiao and Kenneth P. Birman. "A Randomized Error Recovery Algorithm for Reliable Multicast", *IEEE Proceedings Twentieth Annual Joint Conference of the IEEE Computer Societies and Communication Societies (INFOCOM'01)*, pp. 239-248, 2001

[51] A. Striegel, G. Manimaran, "A survey of QoS multicasting issues", *IEEE Communications Magazine,* pp. 82-87, Vol. 40, Issue 6, 2002.

[52] Behrouz A Forouzan, TCP/IP Protocol Suite, *McGraw-Hill Science/Engineering/Math*; 2 edition (June 27, 2002)

[53] Laxman H. Sahasrabuddhe, Biswanath Mukherjee, "Multicast Routing Algorithms and Protocols: A Tutorial", *IEEE Network*, pp. 90-102, 2000.

[54] Military Training Technology [http://www.military-training-technology.com/article.cfm?DocID=2062] *Last accessed January 2008.*

[55] The Simulation Learning company [http://www.3dsolve.com/press_releases/2006_11_16MTT2.html] *Last accessed January 2008.*

[56] Head Mounted Display [http://www.i-glassesstore.com/] *Last accessed January 2008.*

[57] Military and Tactical Operations [http://www.i-glassesstore.com/military-ops.html] *Last accessed January 2008.*

[58] Holloway, Richard and Anselmo Lastra. "Virtual Environments: A Survey of the Technology". *Eurographics '93*: Notes for tutorial TN3. Barcelona, Spain. (Also republished in SIGGRAPH 1994 Course Notes, 17 (pp. A:1- A:36). New York, NY: ACM.)

[59] Chiu D. M., S. Hurst, M. Kadansky and J. Wesley, "TRAM: A Tree-based Reliable Multicast Protocol", *Sun Microsystems Laboratories Technical Report,* SMLI TR-98-66, September 1998.

[60] Logitech [http://www.logitech.com/] *Last accessed January 2008.*

[61] PHANTOM® Omni™ Haptic Device [http://www.sensable.com/haptic-phantom-omni.htm] *Last accessed January 2008.*

[62] PHANTOM® Premium Model Haptic Devices [http://www.sensable.com/haptic-phantom-premium.htm] *Last accessed January 2008.*

[63] PHANTOM Desktop™ Haptic Device [http://www.sensable.com/haptic-phantom-desktop.htm] *Last accessed January 2008.*

[64] PHANToM Premium 6DOF Devices [http://www.sensable.com/haptic-phantom-premium-6dof.htm] *Last accessed January 2008.*

[65] Cyber Glove [http://www.vrealities.com/cyber.html] *Last accessed January 2008.*

[66] Autodesk 3DS Max [http://usa.autodesk.com/adsk/servlet/index?id=5659302&siteID=123112] *Last accessed January 2008.*

[67] Open GL ® [http://www.opengl.org/] *Last accessed January 2008.*

[68] Direct3D [http://www.pluralsight.com/wiki/default.aspx/Craig.DirectX.Direct3DTutorialIndex] *Last accessed January 2008.*

[69] S. Armstrong, A. Freier, and K. Marzullo, "Multicast Transport Protocol," *Internet RFC 1301*, Feb. 1992.

[70] GHOST® Software Development Kit [http://www.est-kl.com/aufbau_general/index_hard_haptic.html?http://www.est-kl.com/hardware/haptic/sensable/ghost.html] *Last accessed January 2008.*

[71] Open Haptics Toolkit [http://www.sensable.com/products-openhaptics-toolkit.htm] *Last accessed January 2008.*

[72] Virtual Reality Modeling System [http://www.w3.org/MarkUp/VRML/] *Last accessed January 2008.*

[73] Moving Picture Experts Group [http://www.mpeg.org/MPEG/index.html] *Last accessed February 2008.*

[74] Distributed Interactive Simulation [http://www.fas.org/man/dod-101/army/docs/astmp/c6/P6C3.htm] *Last accessed February 2008.*

[75] The High Level Architecture [https://www.dmso.mil/public/transition/hla/] *Last accessed February 2008.*

[76] B. Whetten, T. Montgomery, and S. Kaplan, "A High Performance Totally Ordered Multicast Protocol," *Selected Papers from the International Workshop on Theory and Practice in Distributed Systems*, Springer Verlag, pp. 33-57, Vol. 938 Lecture Notes in Computer Science.

[77] Xpress Transport Protocol [http://www.ccii.co.za/products/xtp.html] *Last accessed January 2008.*

[78] Wikipedia Encyclopedia [www.wikipedia.org] *Last accessed February 2008*

[79] K. Miller, K. Robertson, A. Tweedly, M. White, "Starburst Multicast File Transfer Protocol (mftp) Specification," *Internet draft, IETF, draft-miller-mftp-spec-02.txt*, Jan. 1997.

[80] Chiu, D., et al., "TRAM: A Tree-based Reliable Multicast Protocol," Sun Microsystems Laboratory Technical Report, SML TR-98-66, Sun Microsystems, pp. 1-22, Jul. 1998

[81] Henning Schulzrinne, Stephen Casner. "RTP: A Transport Protocol for Real-Time Applications.", *Internet Engineering Task Force, Internet Draft*, October 20, 1993

[82] C. Bormann, J. Ott, H.C. Gehrcke, T. Kerschat, and N. Seifert, "MTP-2: Towards Achieving the S.E.R.O. Properties for Multicast Transport," International Conference on Computer Communications Networks, San Francisco, California, Sep. 1994.

[83] Real Time Control Protocol [http://www.freesoft.org/CIE/RFC/1889/13.htm] *Last accessed in February 2008.*