INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

Design and Implementation of a Multimedia Presentation System Using Real-time Transport Protocol (RTP)

by

Sridevi Palacharla

A thesis submitted to the
Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Engineering

Ottawa-Carleton Institute for Electrical Engineering
Faculty of Engineering
Carleton University
Ottawa, Ontario

May 1997

© 1997, Sridevi Palacharla

National Library
of Canada

Bibliothèque nationale
du Canada

Acquisitions and
Bibliographic Services

Acquisitions et
services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-27004-1

Canada

The undersigned hereby recommended to the Faculty of Graduate
Studies and Research acceptance of the thesis

*Design and Implementation of a Multimedia
Presentation System Using
Real-time Transport Protocol (RTP)*

submitted by Sridevi Palacharla, B.E.(Hons).,
in partial fulfillment of the requirements
for the degree of Master of Engineering.

Dr. A. Karmouch
Thesis Supervisor

Dr. S. A. Mahmoud
Thesis Supervisor

Dr. S. A. Mahmoud
Chair,
Department of Systems and Computer Engineering

Carleton University
June 24, 1997

Abstract

A new class of protocols like RTP, RTSP and RSVP are tailoring today's packet networks for streaming real-time data that once carried only non-real-time traffic. Multimedia networking is now possible with the existing network infrastructure. Media-on-demand services over the Internet is a reality.

This thesis describes a real-time Multimedia Presentation System for media-on-demand over the Internet. It uses RTP and is capable of real-time video and audio transmission. Our system consists of servers that maintain video/audio clips and clients that retrieve and present them to the users over the network. It provides VCR like control features for user interactivity during media playback. The design, implementation and experimental results of our system are presented in this thesis. The system architecture, communication protocols along with media synchronization and dynamic QoS control mechanisms are discussed.

The prototype developed demonstrates that it is possible to transmit real-time audio and video on LAN as well as the Internet. The significance of RTP for future media-on-demand applications is established.

Acknowledgments

I express my gratitude to Professor Ahmed Karmouch and Professor Samy Mahmoud, who acted as joint supervisors, for their valuable advice, encouragement and responsiveness. They have offered me a unique learning experience for which I am thankful.

I wish to thank System Administrator, Narendra Mehta for the support in the laboratory at Carleton University. I also thank the many researchers at MIRL in University of Ottawa for the friendly environment they have created.

I am grateful to my family for their support and understanding. Special thanks to my husband for his encouragement, support and patience.

Table of Contents

Abstract.....	iii
Acknowledgments	iv
List of Tables.....	vii
List of Figures	viii
 Chapter 1 Introduction.....	 1
1.1 Motivation	1
1.2 Thesis Objectives	4
1.3 Thesis Outline	6
1.4 Main Contributions	7
 Chapter 2 Multimedia Information Systems	 9
2.1 Introduction	9
2.2 Distributed Multimedia Information Systems.....	11
2.2.1 Overview	11
2.2.2 Distributed Communication Systems	14
2.2.3 Integrated Media Presentation	16
2.3 DMIS Applications	19
2.3.1 Tele-Learning.....	20
2.3.2 Multimedia News	23
2.3.3 Computer Supported Cooperative Work	25
2.4 Summary.....	27
 Chapter 3 Design of Real-time	
Multimedia Presentation System (MPS)	30
3.1 System Architecture	30
3.1.1 MPS Server.....	35
3.1.2 MPS Client	36
3.1.2.1 Design of GUI	38
3.2 Communication Protocols	40
3.2.1 Session Control Protocol	42
3.2.2 Media Transport and RTP	49
3.2.2.1 Real-time Transport Protocol (RTP)	49
3.2.2.2 Video Transport	54
3.2.2.3 Audio Transport	56
3.2.3 Media Feedback and RTCP.....	58
3.2.3.1 Real-Time Control Protocol (RTCP)	58

3.2.3.2 Media Feedback	60
Chapter 4 Synchronization in a Distributed Environment	61
4.1 Introduction	61
4.2 Synchronization Mechanisms.....	65
4.3 Media Synchronization in MPS	68
4.4 Summary.....	78
Chapter 5 Quality of Service Control	80
5.1 Introduction	80
5.2 QoS Control Mechanisms.....	83
5.3 Dynamic QoS Control in MPS	85
Chapter 6 Implementation	89
6.1 Hardware and Software Environment	89
6.2 Software Architecture	90
6.2.1 MPS Server	92
6.2.2 MPS Client	95
6.3 User Interface and Mode of Operation	98
6.4 Experimental Results.....	102
6.4.1 Experiments over LAN and Analysis.....	105
6.4.2 Experiments over Internet and Analysis.....	110
Chapter 7 Conclusions.....	116
7.1 Summary.....	116
7.2 Future Work	118
Appendices	121
A Publications.....	121
Bibliography	122

List of Tables

Table 2.1 Multimedia Storage and Presentation Requirements.....	10
Table 3.1 Error Classes and Codes in Session Control Protocol.....	46
Table 3.2 Messages in Session Control Protocol.....	47
Table 3.3 Payload Types for Standard Audio and Video Encodings.....	53
Table 4.1 Quality of Service for Synchronization Purposes.....	64
Table 5.1 Quality of Service for Media Object Presentation.....	82
Table 6.1 Video and Audio Test Data Details.....	103
Table 6.2 Hosts and their Functions in our Experiments.....	104
Table 6.3 Video Test Results on LAN.....	106
Table 6.4 Video Test Results over Internet.....	112
Table 6.5 Audio Test Results over Internet	112

List of Figures

Figure 2.1 Distributed Multimedia Information System Architecture.....	12
Figure 2.2 OSI Reference Model Layers in Packet Switching.....	15
Figure 2.3 General Architecture of a Multimedia Learning System.....	22
Figure 2.4 Architecture of a Multimedia News System.....	24
Figure 2.5 A Multimedia Collaborative System Architecture.....	26
Figure 3.1 Global System View.....	32
Figure 3.2 Real-time Multimedia Presentation System Architecture.....	33
Figure 3.3 Layout of Graphical User Interface in MPS.....	39
Figure 3.4 Communication Protocol Layers in MPS.....	41
Figure 3.5 Session Control Protocol Message Structure	43
Figure 3.6 MPS Server State Transition Diagram.....	48
Figure 3.7 RTP and the Protocol Stack	50
Figure 3.8 RTP Fixed Header.....	52
Figure 3.9 JPEG Header	54
Figure 3.10 Video Packetization in MPS.....	55
Figure 3.11 Audio Packetization in MPS.....	57
Figure 3.12 RTCP Header.....	59
Figure 3.13 RTCP Sender Report (SR) Information	59
Figure 3.14 RTCP Receiver Report (RR) Information.....	60
Figure 4.1 Typical Temporal Relationships.....	62

Figure 4.2 Data Location Models for Distributed Systems.....	63
Figure 4.3 End-to-End Delay Variation at the Receiver.....	69
Figure 4.4 Association Between Audio and Video Frames.....	70
Figure 4.5 Estimation of Inter-Arrival Jitter in Audio Packets.....	72
Figure 4.6 Estimation of Inter-Arrival Jitter in Video Frames.....	73
Figure 4.7 Audio and Video RTP Timestamp Generation.....	74
Figure 4.8 Synchronization Mechanism for the Case of Audio Only.....	74
Figure 4.9 Synchronization Mechanism for the Case of Video Only.....	75
Figure 4.10 Synchronization Mechanism for the Case of Audio and Video	77
Figure 5.1 QoS-Layered Model for Multimedia Communicating System.....	81
Figure 5.2 Packet Loss Estimation at the Receiver	85
Figure 5.3 QoS Control Steps in MPS	86
Figure 6.1 Layout of Workstations in MPS	89
Figure 6.2 Algorithm for Concurrent MPS Server Creation.....	92
Figure 6.3 Process Oriented Structure of MPS Server	93
Figure 6.4 Class Hierarchy in Slave Server Process (SSP).....	94
Figure 6.5 Class Hierarchy in Audio Server Process (ASP).....	94
Figure 6.6 Algorithm for MPS Client Creation.....	95
Figure 6.7 Process Oriented Structure of MPS Client.....	96
Figure 6.8 Class Hierarchy in Master Client Process (MCP).....	97
Figure 6.9 Class Hierarchy in Audio Client Process (ACP).....	97
Figure 6.10 User Interface of MPS.....	99

Figure 6.11 A Snapshot of MPS Showing Cnn Video Clip.....	101
Figure 6.12 Interaction Sequence Diagram for Audio Only Scenario.....	102
Figure 6.13 Decode and Render Times for Cbc M-JPEG Video Frames	104
Figure 6.14 Relative Transit Delay for Cbc Video Frames on LAN.....	107
Figure 6.15 Inter-Frame-Arrival Jitter for Cbc Video on LAN.....	108
Figure 6.16 Relative Transit Delay for Cbc Audio Packets on LAN.....	109
Figure 6.17 Inter-Arrival Jitter for Cbc Audio Packets on LAN.....	110
Figure 6.18 Results of Traceroute fom Vanguard to Rigel.....	111
Figure 6.19 Relative Transit Delay for Cbc Video Frames over Internet.....	113
Figure 6.20 Inter-Frame-Arrival Jitter for Cbc Video over Internet.....	113
Figure 6.21 Relative Transit Delay for Cbc Audio Packets over Internet.....	114
Figure 6.22 Inter-Arrival Jitter for Cbc Audio Packets over Internet.....	115

Chapter 1

Introduction

1.1 Motivation

Over the past few years, the advent of powerful computational technology coupled with broadband network technology has spurred much interest in multimedia research. Multimedia communications has revolutionized the way people communicate by supporting the combination of not only text, graphics and images but also audio and video. Multimedia research has spanned various fields including media specification, integration, scheduling and delivery protocols [LIT90, STE90, EME93, ROD95, BLA96]. Several issues in these related fields have been addressed in a distributed multimedia information system and some have been experimented using Local Area Networks (LAN) or high-speed Asynchronous Transfer Mode (ATM) networks [LIT92, RAN92, MIL93, RAV93]. At present ATM networks are expensive to deploy over wide areas and most ATM vendors still do not have heterogeneous solutions for interoperability. In order to make multimedia communications practical and easily available among widely distributed users, much research is required to enable existing packet-networks such as the Internet to carry multimedia data.

In the past packet-switching networks were designed for reliable rather than timely delivery of information for traditional applications that are tolerant to reasonable and variable network transit delays. New generation of multimedia applications involve real-time audio and video delivery. This presents a new challenge of continuous and timely delivery. Multimedia networking places significant bandwidth demands and Quality-of-Service (QoS) requirements on the existing networks. Although at first it appears that packet-switched network technologies can not meet these continuous demands, sophisticated and enhanced protocols to work along with the existing network protocols can make multimedia applications a reality over these networks.

Today's packet-switched networks can handle real-time traffic along with non-real-time traffic with several key additions. A first addition is an optimized transport protocol for real-time data unlike Transport Control Protocol (TCP) which was developed for reliable data transfer. With video streams, unlike file transfers, reliability can be traded with real-time requirements. Occasionally lost packets can be compensated without significant visual effect to the user during video presentation. As video data is highly voluminous this can easily exhaust the available bandwidth resources. Second, bandwidth requirements need to be reduced by support of compression techniques and multicast. Since resources are still scarce to meet the requirements of multimedia applications, a third addition is the ability to reserve network resources such as bandwidth through a class of traffic priorities to achieve the required QoS. In addition to these enhancements to the network infrastructure the end-nodes must be able to handle compression, decompression and

synchronization of various network streams. Technologies such as ATM can improve the situation by providing more bandwidth once all these elements are in place.

Recently, multimedia applications in the areas of education, business and entertainment are gaining more popularity in the Internet community. The information superhighway is gearing to serve as a vehicle for the exchange of rich multimedia data such as audio and video along with traditional data types such as text and still images. The Internet research community has produced a new generation of protocols [DEE91, SCH96a, ZHA93, RAO96] for use in real-time multimedia applications over the Internet. IP multicast has been the source of inspiration for the generation of these protocols to enhance the existing packet-switched networks for real-time multimedia data.

MBone or multicast backbone is a subset of the Internet which uses enhanced protocol standards for multicast and real-time support of multimedia data. It has been the test-bed for multimedia applications such as audio conference tool, *vat* [JAC92], video conference tool, *vic* [MCA95], Network Video tool, *nv* [FRE] and many more. Some of these applications use Real-time Transport Protocol (RTP) [SCH96a]. These are primarily conferencing applications. However not much research in the light of these new generation of protocols has been reported on media-on-demand services over the Internet [CHE96].

MEDIABASE project was created by Ottawa University's Multimedia Information Systems and Research Laboratory to study various aspects of multimedia information systems. The goal of MEDIABASE is to design an advanced high performance

multimedia information and communications system with a particular focus on document architectures, database models, high-level communication and synchronization, and real-time physical storage of multimedia data . In order to conduct a variety of experiments and to study these issues a Distributed Multimedia Information System (DMIS) was created in MEDIABASE [MEG94a]. Media is produced at the production server and stored in multiple media servers comprising a distributed multimedia database. This information is later retrieved over the network by various clients.

In MEDIABASE project DMIS uses multimedia document for information exchange which is based on MEDIADOC architecture [EME93]. Prior to this work several experiments were conducted over LAN and ATM networks using proprietary protocols designed for media transfer over User Datagram Protocol (UDP) and TCP [MEG94b, ROD95]. With the availability of standard protocols for real-time transfer of audio and video such as Real-time Transport Protocol (RTP), this thesis investigates media-on-demand issues using RTP both over LAN and Internet with a particular focus on audio and video media types.

1.2 Thesis Objectives

The main objective of this thesis was to design and implement a real-time Multimedia Presentation System (MPS) to handle the transport and end-to-end synchronized playback of multimedia clips consisting of audio and video. The system had to be designed based on object-oriented design principles to support interactive playback with VCR like control

functions and support extensions to other media types. The user has the option even to select the type of media for playback. This is required as the user can at least receive audio if not both video and audio. This is especially important in the Internet scenario as the resources are not often abundantly available to have enough bandwidth for both audio and video transfer. The system had to use independent data paths handled by different processes for different media types as opposed to interlacing both media types on a single path and handled by a single process. This leaves enough scope for possible extensions to incorporate other media types without disturbing the existing modules. Object-oriented design also enables extensions to different compression standards.

The system was also designed to operate within a distributed environment. Therefore, the system had to be robust and scaleable in order to handle the retrieval of media over computer networks. Not only synchronization is of importance but also user QoS specification and dynamic QoS control. Since media is retrieved over networks and as the QoS guarantees are not readily available, the system has to detect current network conditions and dynamically adapt to these variations. The thesis addresses the scenario of unicast transmission of pre-recorded and stored audio and video data.

Thus the main objective of the thesis can be summarized as design and implementation of a scalable, interactive real-time multimedia presentation system with support for real-time transmission of data through the use of standard protocols and object-oriented design principles.

1.3 Thesis Outline

The next chapter will review Distributed Multimedia Information Systems (DMIS). In this chapter, a new class of adaptive multimedia applications, tele-learning, news-on-demand and groupware applications are described. The common requirements of these applications are identified. To meet these requirements client-server architectures, standard protocols for media transfer and resource reservation are presented.

Chapter 3 describes the design of real-time Multimedia Presentation System (MPS). In this chapter, the design issues, system architecture and communication protocols used in MPS are described. Chapter 4 discusses the issues related to media synchronization. An algorithm is described to achieve inter-media synchronization between audio and video data streams. Chapter 5 reviews issues related to QoS and describes QoS control mechanisms used in our system.

Chapter 6 gives the implementation details of MPS. The implementation allows for audio and video session control along with user-interactivity. It allows for continuous transfer of multimedia data over a communication network and is capable of dynamic QoS adaptation. It also discusses the experimental results in LAN and Internet environments. The results presented include cumulative statistics of audio and video transfer over different network infrastructures and delay jitter estimations in the face of varying network

conditions. Chapter 7 concludes the thesis with future extensions to our system and further research issues.

1.4 Main Contributions

The main contribution of this research is the design and implementation of a real-time Multimedia Presentation System (MPS) capable of synchronizing audio and video independent of the underlying network infrastructure. This system differs from other presentation systems because it uses RTP for audio and video transfer and hence uses separate data path and control path for each media instead of interleaving both in a single path. The system uses client-server architecture to work in a distributed environment. The system uses RTCP, a control protocol of RTP to exchange control information on a per media basis. This allows to keep track of reception quality and adapt QoS to changes in network.

A session control protocol was designed for exchange of control information between the client and server session managers. Other contributions of the work performed for the completion of this thesis include a friendly utility to create frame indices for huge motion JPEG (Joint Photographic Experts Group) created using SunVideo card. This facilitates random access to frames instead of linear sequential searching. Other utilities include a conversion utility to convert from motion JPEG to JPEG, a print utility to print the total number of frames in a huge motion JPEG file to standard output. A graphical user

interface, called AV Player was created as the client application for the media presentation system.

The results of the experiments conducted on LAN and on the Internet are presented. The percentage of packets and percentage of frames lost are measured based on the RTP header information. Relative transit delay and inter-arrival jitter of audio frames is also estimated based on the RTP information. Similarly for video, relative transit delay and application-level inter-frame-arrival jitter is estimated to study the nature of the various network infrastructures and to analyze the real-time traffic over these networks.

Chapter 2

Multimedia Information Systems

2.1 Introduction

Multimedia Information System (MIS) results from the merging of the computing, communications and broadcasting industries. This merging is possible due to the technical advances in high-speed broadband networks, powerful desktop computers, workstations and information storage and compression techniques. Multimedia information systems aim to improve the way people communicate by supporting communication of various types of media and thereby creating a more “natural” way to exchange information [KAR93].

The media types which must be supported by the MIS include the discrete media types such as text, image and graphics as well as the continuous media types such as audio, video and animation. Table 2.1 lists these media types and compares their storage and presentation requirements. As can be seen from the table, the storage and presentation of the different media vary greatly with respect to one another. In order to establish multimedia communications, the MIS will have to satisfy the full gamut of services required by these diverse media.

In general, most communications occur between individuals or groups which may be separated in space and time. In order to become the de facto communication standard of the future, MIS must be able to handle the most general case of information exchange. That is, MIS must be capable of communicating information which may be stored or created live between communicating entities within a distributed environment. This leads to the more generalized distributed multimedia information system (DMIS).

Table 2.1: Multimedia Storage and Presentation Requirements*

Media	Data Type	Storage	Presentation
Text	- ASCII - Formatted ASCII (i.e. postscript)	- approx. 2 kB/page	- static visual display - 0 B/s
Image	- Bitmapped Graphics - colour: 1-24 bit - size: iconic-full screen	- 64 kB - 75 MB/image	- static visual display - 0 B/s
Audio	- coded (or non-coded) digitized stream of voice (8 kHz) or CD (44.1 kHz) quality sound	- voice: 6-44 kB/s of audio data - CD: 176 kB/s of audio data	- continuous aural playback - voice playback rate: 6-44 kB/s - CD playback rate: 176 kB/s
Animation	- synchronized stream of images	- 2.5 MB/s of 16 bit colour, 320X240 pixels/frame, 16 frames/s	- continuous visual playback - playback rate: 2.5 MB/s
Video	- TV analog or digital stream of images	- 27.2 MB/s of 24 bit colour, 640X480 pixels/frame, 30 frames/s	- continuous visual playback - playback rate: 27.7 MB/s

*Note: data from [FUR94].

This chapter is an introduction to distributed multimedia information systems. The next section will discuss distributed systems in general along with the distribution, integration and presentation issues of the media that these systems entail. Section 2.3 will then present some applications which are built upon distributed multimedia information systems; particularly, tele-learning, multimedia news-on-demand and multimedia groupware.

2.2 Distributed Multimedia Information Systems

2.2.1 Overview

The distributed multimedia information system (DMIS) communicates multimedia data between entities which may be both geographically and temporally dispersed. Figure 2.1 demonstrates a distributed multimedia information system, composed of various user-sites with different applications, information repositories among different servers distributed across the network; possibly different computer networks. There are three fundamental elements to DMISs in general. These are the source, presentation and transport elements required to manipulate the multimedia information within the system.

The source element deals with the origination of the multimedia information. This includes the production and the storage of multimedia information. Media production can be handled by media editors like text and image editors or by using computer digital

technology to scan images and store the digitized image content. Audio and video can be produced using camera, microphone and a special hardware card which digitizes the content. In any case, media data can be produced for either immediate consumption and discarding, thereby creating live sources, or for future retrievals and therefore requiring some form of storage for the media data.

Figure 2.1: Distributed Multimedia Information System Architecture

Storage deals with the persistence of the media information as well as information security, versioning, aging and volume management. Multimedia databases and file servers are responsible for managing the media information which will be stored electronically in the form of files or documents. In general, electronic files and documents are physically

stored on the computer's hard disk. However, table 2.1 demonstrated that multimedia data can require enormous amounts of storage capacity. For instance a 30 minute audio/video clip will require approximately 50 GB of the computer's hard disk. Since most computer systems have limited primary storage capacity and because of the sheer volume of data typically found in multimedia information systems, storage servers based on secondary memory systems like tape drives, optical disks or "juke boxes" will also become necessary for most applications. Sophisticated compression techniques like JPEG (Joint Photographic Experts Group) and MPEG (Moving Picture Experts Group), which are capable of achieving 10:1 to 50:1 and 30:1 to 100:1 compression ratios respectively, will also be used to reduce the storage burden of video data.

The presentation element deals with communicating the multimedia information at the user's local workstation. It is within the presentation element that the human/machine interaction occurs and therefore, the most important aspect of the local presentation is the user interface. For multimedia applications, graphical user-interfaces (GUIs) must be designed so that they provide powerful semantics for the coherent interaction between the user and the various types of media. This interaction must be independent of the origin of the media (local disk, remote live source, compressed images, tape drives, etc.) and must offer a natural means to interface with the user. Hence, it is essential for the presentation element to offer graphical user interfaces which are serendipitous and where the media appear to be ubiquitous within the distributed multimedia information system.

The transport element deals with the communication of the multimedia data throughout the DMIS. For multimedia data, robust, real-time transport protocols are of prime importance. These protocols must be flexible enough to support widely diverse transport requirements: from relatively small, bursty transfers of text, image and graphics data; to large continuous transfers needed for audio and video media types. These transport protocols must also be able to dynamically adjust to the changes of service quality caused by the non-deterministic nature of communication systems and by the diversity of the communication components which form a distributed multimedia information system.

Distributed multimedia information systems are characterized by heterogeneity. There are two sources to heterogeneity in DMISs. The first is caused by the very nature of multimedia data, i.e. the diversity of the media types; and the second source is the variety of computer and communication equipment linked together in typical distributed communication systems. Discussions on distributed communication systems and integrated media presentation provided next suggest solutions to the problem of heterogeneity in DMISs.

2.2.2 Distributed Communication Systems

The successful realization of distributed communication systems is primarily due to the implementation of layered architectures and client-server modeling. The client-server paradigm describes a communication model between the service consumers or clients, and

the service providers or servers. By allowing servers to become clients to other servers, complex services can be built by combining simpler services.

Figure 2.2: OSI Reference Model Layers in Packet Switching

Layered architectures are the result of hierarchically grouping services according to their complexities and functionalities. Entities in one layer request services from entities in the layer which is immediately beneath them and provide services to the entities in the next higher layer of the hierarchy. The OSI (Open Systems Interconnection) reference model is a computer communication architectural model which is used to standardize communication protocols so that various computer systems may connect with one another. There are seven layers to OSI model; they are the physical, data link, network, transport, session, presentation and application layers. The lower layers control single link, node to node bit and packet transmission, while the next layers control the end to end packet transmission which provide basic communication services to the upper application layers. Figure 2.2 shows the OSI reference model layers used in packet switching.

The client-server and OSI reference models provide a powerful architecture which simplifies the design, implementation and maintenance of large communication systems. The standard layer interfaces allow designers to plug components into their respective layers and to use the services offered by these components. Therefore the client-server model maximizes re-usability and inter-connectivity and facilitates the implementation of common services for the whole system independently of any of the constituent communicating entities which make up the communication system.

2.2.3 Integrated Media Presentation

The aim of multimedia applications is to be able to efficiently communicate various types of media. Because each media type has very different manipulation and presentation requirements, the multimedia information system must provide support for media integration. While media integration is by itself a challenge in the design of multimedia information system, there are also other major challenges pertaining to the delivery of media data and integrated presentation.

Media integration combines the various media elements so that they can be manipulated and presented by the multimedia information system. There are two types of media integration required by the multimedia information systems : spatial and temporal. Spatial integration groups media elements according to their spatial relationships. Such relationships include position, dimension, orientation and any information pertinent to the

visual representation of the media. Temporal integration deals with the timing relationships or the synchronization needed to render the media at the local workstation. These relationships include start time, duration, playback rates and any information required to support real-time concurrent or sequential media presentation.

There are two aspects to the problem of multimedia synchronization: definition and delivery [STE92]. Definition deals with the specification of the intra-media and the inter-media relationships of multimedia data. Definition encompasses the set of language constructs, hyper-links, scripts and document architectures needed to integrate the multimedia information at the logical level.

Delivery deals with the transport of the media information according to their spatial and temporal relationships. It is the delivery system which schedules the media retrievals from the server(s). Presentation controls their synchronized playbacks at the client and determines the output layout for the media information as per the media integration specification. Together the integrated media presentation system is the set of algorithms, protocols and communicating entities which enforce the policies of the synchronization definition from the source to destination.

Many systems use the concept of the multimedia document [EME93, REI94] to address media integration definition. In general, multimedia documents provide architectures capable of representing the spatial and temporal relationships between the document's constituent media elements. The advantage of the multimedia document is that it facilitates

application design and integration by using the single document as the vehicle for the information interchange of media elements composing complex relationships.

In order to handle media delivery in a distributed multimedia information system there arises a chain of key requirements such as efficient usage of available network bandwidth, support for data compression, QoS guarantees, use of standard communication protocols for media transport, and synchronization of multiple media streams to present them according to the media integration definition.

Efficient usage of network bandwidth can be achieved by supporting multicast capability and use of data compression standards. Multicast technology [DEE91] is useful in scenarios where a group of receivers have a common interest in receiving data originating from the same sender. Multicast exploits this interest by enabling to send a single copy to the group rather than sending a copy to each of the receivers. Video is highly bulky and video compression standards such as JPEG and MPEG should be supported to achieve efficient bandwidth usage. This is especially required for Internet usage.

Assured QoS can give an estimate of upper bounds on end-to-end delay which is important in media presentation, concurrent or sequential. Since in a DMIS multimedia data traverses from a source to destination across a network, random delays are inevitable without a sophisticated support for reservation of the required bandwidth which mitigates randomness and assures delay bounds. For routers, the Internet Engineering Task Force (IETF) has developed the ReSource reserVation Protocol (RSVP), as end-to-

end internetwork protocol that allows applications to dynamically reserve necessary network resources (e.g., buffer resources or bandwidth) for different classes of service.

Use of standard protocols is necessary to attain communication among the various entities of a DMIS. For the delivery of real-time media such as audio and video data types there is a requirement of real-time data transport protocol. This protocol should lend itself to tailor to the needs of the application rather than the application tailor as per the protocol. The timing information needed to achieve synchronization should also be supported. Since resources reservation is possible and not yet abundantly available, support for QoS monitoring and control by the transport protocol on an end-to-end basis is still very much desired. As a part of IETF effort, the Real-time Transport Protocol (RTP) has evolved that caters to these needs.

2.3 DMIS Applications

There exists many application domains for which distributed multimedia information systems are required. These application domains include: training, education, simulation, publishing, public information, groupware, sales, advertising, entertainment, as well as future applications which are yet to evolve as distributed multimedia information systems. In this section, three DMIS applications will be reviewed so that their common components may be identified as base requirements for distributed multimedia information systems in general. The applications are tele-learning, multimedia news and computer supported collaborative work (CSCW).

2.3.1 Tele-Learning

The usefulness of computers in education has already been widely recognized, even before the arrival of multimedia technology. However, in the past, the benefit brought by computer systems was mainly due to their ability to efficiently perform time-consuming, repetitive administrative and clerical tasks. Although testing and monitoring were easily performed by computer systems, they were seldom used for teaching purposes because they offered no distinct advantage over conventional methods. The advent of multimedia technology, however has renewed interest in computer based learning [HEP94]. The main reasons for this renewed interest is because distributed multimedia information systems can enhance computer learning systems by improving information sharing and information reachability.

Multimedia learning systems are typically composed of multimedia production center, database and user workstations, all of which are connected by a computer network as shown in figure 2.3. The production center is used to digitize media and create the documents which will be used for the development of courseware. These documents and digitized media are stored in the multimedia database. Students and teachers are connected to the computer network from workstations which may be in classrooms, offices, labs and even their own homes. The general case is where the lectures are pre-recorded and later retrieved but the other case where lectures or seminars may be live, should also be

supported by such a system. In live lectures, a group of students may be interested in attending it remotely.

Tele-learning applications can increase information sharing because of the multimedia database which is inherent to these systems. This database allows many users to access the same information through the computer network thereby increasing the distribution of information to local and remote users. The database also has a greater potential for information re-use, since files and documents which may have been created for one course may be copied wholly or in part for the creation of other course materials. For instance, a course document which was designed to teach group interactions for the psychology department may also be used to teach team management for the administrative department.

Multimedia learning applications can also increase course material production. Because the production center of the distributed multimedia information system permits educators to digitize images, sound-tracks and video-tracks, these systems can thereby remove the dependency on expensive material resources. For example, a lesson on the modern art works of the National Gallery could be created by digitizing and presenting only that portion of a video filmed at the gallery. Educators gain two-fold from this use of digital computer technology. Firstly, educators can now create shorter and more specific sequences of information from larger generalized media sources, and secondly, educators can use materials which result from their own personal experiences. For instance, the video could have been taken during the teacher's last vacation or field trip at the gallery.

Figure 2.3: General Architecture of a Multimedia Learning System

Multimedia learning applications can improve information reachability because they provide support for a wider choice of media than conventional teaching methods. Course materials communicated through text, image, audio and video formats are absorbed with varying degrees of efficiency by different viewers. Therefore, by designing course material in as many formats as possible, the reachability of the information is optimized for all potential viewers.

The most interesting innovation brought to education by multimedia technology is learning through computer simulation. Simulations permit users to experience events and learn from these experiences. Although no simulation can provide a full or a thorough

experience, partial experience is better than none. Simulations also allow immediate explanation and interpretation of experienced events, which is not always the case for real-life situations. Through experience and explanation, multimedia simulations help create a unique learning environment.

2.3.2 Multimedia News

Multimedia news is an initiative which aims to provide news-on-demand to satisfy the information requirements of a highly diversified client-base [MIL93]. The multimedia news system exhibits many advantages over the news services currently available. In effect, multimedia news systems built on multimedia databases can cater to specific client interests. Clients create a profile to register their current interests with their news providers which will then communicate only the information matching the profile. The advantage of the profile is that it permits the client to receive specific news stories without eliminating the possibility of later retrieval of other information from the multimedia news database. The profile also reduces the burden involved in distributing information conceived to satisfy a general client base.

Another advantage of multimedia news systems is their ability to communicate time-critical information. A change in the interest rate announced by the central bank is an example of a time-critical news event. Investors may immediately react to the change and modify their portfolios accordingly. If investors receive the announcement too late, it could represent a significant loss of value of their portfolio.

Multimedia news systems are also more efficient than current news systems when it comes to information management. Information management includes the production, distribution and storage of news segments. Through digital computer technology, news providers will be able to consolidate the production of video, audio and printed news stories within a single news production center (see figure 2.4). Distribution can be geared to satisfying the needs of individual clients as opposed to satisfying printing and

Figure 2.4: Architecture for a Multimedia News System

broadcasting deadlines. Distributed multimedia databases using broadcast, multi-cast and point-to-point communication protocols will provide the support for client profiles

requesting news alerts, story updates, prioritized information delivery or even regular information multicasts.

2.3.3 Computer Supported Cooperative Work

Computer supported cooperative work (CSCW) or “groupware” aims to improve the way groups of people collaborate through better use of computer and communication technologies. Consulting, video conferencing, joint editing, project planning and management, are a few examples of CSCW applications. In all these applications, the group collaborates to achieve a common goal. In joint editing for instance, the group’s common goal is to produce a document. While one member creates an image for the document, another prepares the textual information to accompany the picture.

There are several aspects to consider in order to successfully realize CSCW applications. These aspects include multi-user interfaces and collaboration strategies, protocols for synchronous and asynchronous communications and the sharing of data, services and resources within a distributed environment. Because many of these aspects also apply to distributed multimedia information systems, DMISs provide a solid foundation from which cooperative multimedia applications can evolve.

In general, cooperative multimedia uses the concept of a multimedia document to organize, represent and structure information exchanged between remote cooperating agents [KAR93]. For example, the multimedia document is the end goal of joint editing

applications; project schedules and progress reports are created by project planning and management applications; and even video conferencing can use the concept of the multimedia document by recording the audio, visual and data information into the minutes of the conference.

Figure 2.5: A Multimedia Collaborative System Architecture

CSCW applications will potentially increase productivity and efficiency of group collaborations. DMISs will provide the backbone for CSCW applications and help create an ideal communication environment for representing ideas in a greater number of ways. CSCW applications will help reduce the expense and time required for members to organize and travel to their meetings. CSCW applications will also help reduce the cost

and time associated with the production and revision of information by making use of digital computer and database technologies.

2.4 Summary

The ability to exchange various types of media within widely diversified information networks will create a new communication environment with a greater potential for information reachability. This new communication environment is the result of the distributed multimedia information system (DMIS). Not only do DMISs enhance the way information is represented, but they also improve the way this information is communicated. DMISs can increase information sharing, across geographical and temporal boundaries and they can also help create a global interactive communication environment for all types of media information.

In this chapter, we have reviewed distributed multimedia information systems in general, as well as potential applications for these systems. From this review, several components which are common to all DMISs have been identified. These components include production centers, storage centers, user sites and computer communication networks. Technologies for the production and storage of computer information, as well as for communication to remote users through computer networks are evolving to overcome the complexities brought on by the large, real-time nature of multimedia information.

Two problems were identified as major obstacles to the successful realization of distributed multimedia information systems. The first problem is due to the distributed nature of distributed communication systems and services. Client-server architectures were presented as a means to overcome this problem. By breaking down services into functional layers, system implementation is made more manageable by the re-use of peer services via standard protocols.

The second problem identified is integrated media presentation. Media integration combines the various types of media so that they can be stored, manipulated, communicated and presented coherently by the distributed multimedia information system. Of the two types of media integration needed, only temporal integration, or synchronization, is needed for the production, storage, network and user interface components of the distributed multimedia information system.

Media presentation requires several key requirements to be addressed to achieve synchronized presentation. The multimedia synchronization problem is broken down into two parts: synchronization definition and synchronized delivery. Multimedia document architectures provide a means to implement the definition aspect. However synchronized delivery which leads to actual physical integration of multimedia data will be controlled by a set of services which make up the synchronized multimedia delivery system.

The use of compression standards and multicast technology improve bandwidth usage, while resource reservation protocols such as RSVP allow applications to dynamically

reserve network resources, RTP addresses QoS monitoring, timing information required for media synchronization and supports multicast. These new services can meet the general requirements of DMIS applications such as tele-learning, news-on-demand and CSCW.

Chapter 3

Design of Real-time Multimedia Presentation System (MPS)

3.1 System Architecture

The real-time Multimedia Presentation System (MPS) was designed as a media-on-demand tool to experiment with issues concerning real-time media transport between widely distributed entities such as over the Internet. The general requirements of such a tool include support for the following :

- **Real-time media transmission:** The Internet was designed as data communications network with no strict delay bounds for the data that is carried. In order to achieve real-time data communications over the same network, sophisticated protocols are required to carry audio and video over often lossy and variable delay prone network.
- **Scalability:** The number of Internet users is growing day by day on a world-wide basis. Any system built with Internet in mind should therefore be scalable. The system should

perform over a LAN covering few users as well as over wider networks covering many users as on the Internet.

- **Flexibility/Open System:** The system should be flexible to be a part of an integrated system, so that it can be combined with the existing or future Internet navigation tools. The system should be based on a common protocol to be used for stored or live media delivery. Today's media-on-demand system should not only support pre-recorded data but also possible live data in future. An example is to be used in live lectures-on-demand. This needs support of standard protocols and media encoding formats.
- **User interactivity:** This is required to support VCR like functions such as play, stop, pause, resume, forward and rewind controls to the user in the case of pre-recorded data. For live data record, play and stop controls can be supported.

The requirements that are also important to be supported by the system are as follows:

- **Media Synchronization:** Synchronization is of utmost importance in any multimedia system. In our system we deal with audio and video media types. Both intra and inter media synchronization are to be maintained; details are given in chapter 4.
- **Quality-of-Service (QoS) adaptation:** In order to maintain the synchronization between various media types QoS parameters such as end-to-end delay bound and bandwidth are required to be guaranteed. Since resources to make such guarantees are not readily

available for an Internet user and are expensive, we need mechanisms so that the system can dynamically adapt to varying network conditions.

The system design of MPS is based on client-server paradigm and object-oriented

Figure 3.1: Global System View

design principles. The client-server paradigm is a concept for describing communications between service consumers (clients) and service providers (servers). This paradigm is suitable for our needs of request-response oriented communication; the client, requesting for a service and a server responding to the client's request. A system based on object-oriented design framework permits to easily incorporate additional modules or to derive from existing modules. Incremental development can be achieved, thus making enhancements less cumbersome. In the client-server architecture of MPS there can be

multiple servers and multiple clients communicating over a network (figure 3.1). A single server can support multiple clients simultaneously. The architecture of MPS with a single client, single server and their components is shown in figure 3.2.

Figure 3.2: Real-time Multimedia Presentation System Architecture

As shown in figure 3.2 data flow is represented using a solid line and control flow is shown using a dotted line. The direction of flow is indicated by the arrow heads. The communication protocols used for data delivery and exchange of control information between various components are detailed in section 3.2. The user requests media clips through the graphical user-interface (GUI) provided at the MPS client. The client retrieves media data from the MPS server and presents the data to the user. The dotted line

between server Session Control Manager (SCM) and client SCM represents the flow of control messages for the corresponding session. The dotted lines between the RX and TX media agents indicate a feedback channel to report reception quality and control media delivery rate respectively. The components are described in more detail in the following sub-sections.

Each type of media is handled by a separate media agent both at the server and the client. There are several advantages of this design. Since audio and video data are delivered using separate TX agents, distribution of media and distribution of TX agents among different servers can be easily achieved. The client can request audio from one server and video from a different server. The separation also permits priorities to be set at all levels along the path of the media flow from the server to the client. For instance, it allows to set higher priority in terms of scheduling to audio media agents at the source and destination compared to video media agents as the compute-intensive video processing should not degrade aural quality. By carrying each media type as a separate stream, priorities at the network level can also be assigned with little processing. Another advantage is system flexibility can be attained, so that new media types other than audio and video can also be supported by adding new media agents without disturbing the existing agents.

3.1.1 MPS Server

The main tasks of the server are as follows:

- Accept and enforce concurrent client requests for media playback and control.
- Deliver each media type to the client in real-time.
- Monitor media reception quality and take appropriate action.

The server design follows the traditional method to support concurrent client requests by having a master server that listens for incoming client connection requests and hands each client to a slave server [COM93]. Each slave server carries out further client requests until the client has finished all its requests. In figure 3.2, the server MPS reflects the slave server that actually carries out the client requests.

The MPS server is composed of the following main components :

- Session Control Manager (SCM).
- Audio transmission agent (audio TX agent).
- Video transmission agent (video TX agent).
- Audio-Video data-store (AV data-store).

The client requests are handled by the SCM component at the server. The SCM parses the request and initiates and controls the TX agents. The SCM informs the appropriate TX agent to start transmitting media data, pause, resume or stop sending the media data of a particular clip as per the requests of the client.

The TX agents are responsible for reading the media data from AV data-store and notifying the SCM of the server of any error and abnormal conditions. These errors are in turn notified to the client by the server SCM. The TX agents also perform QoS control by monitoring the reception quality of each media. This is especially important for video TX agent since it requires more bandwidth than audio TX agent. The video TX agent modifies the frame rate of video delivery based on the reception quality feedback.

AV data-store consists of digitized and compressed video files and audio files. For video we use a series of JPEG [PEN93], still images which is commonly known as motion-JPEG or m-JPEG. Currently, motion-JPEG files are supported for video and 8-bit 8 kHz μ -law audio files are supported for audio.

3.1.2 MPS Client

The client performs more complex tasks compared to the server. The following are its main tasks:

- Support controls similar to a VCR for media playback.
- Interpret user commands and send requests to the server; receive server responses and communicate them back to the user.
- Receive and reassemble media data.
- Decompress video data.

- Play out media data synchronously.
- Provide QoS feedback.

Achieving synchronized media playback is the most important task and involves coordination of several components in the system. The issues of synchronization and QoS feedback and control are detailed in the next two chapters.

The main components of MPS client are as follows:

- Session Control Manager (SCM).
- Audio reception agent (audio RX agent).
- Video reception agent (video RX agent).
- Graphical User Interface (GUI).

In addition to these the client also has the video display component and a speaker for audio output.

The user indicates various choices through the GUI. The important selections include the title of the clip and the type of media pertaining to that clip. The user can request only audio, only video or both audio and video data of a clip. Design details of the GUI are discussed in the next sub-section.

The client SCM initiates the RX agents as a first step of a session and controls their actions during the session. It communicates the crucial information for delivery of media data, the RX agents' receiving port numbers, to the server SCM as a first request message. At any point, errors that the client SCM may receive from the server or from the

RX agents are indicated to the user through the GUI. It interprets the user interactions through the GUI and sends appropriate request messages to the server SCM. These requests primarily reflect the user's choice to start playing the clip, pause, resume, forward, rewind or stop playing.

The audio and video RX agents are responsible for reassembling the received media data, decoding if necessary and rendering to appropriate devices. Video RX agent thus has an additional responsibility of decoding video data. The RX agents report any error or abnormal conditions to the client SCM. The RX agents have the task of performing synchronization during playback. Each RX agent can independently achieve intra-media synchronization by accounting for network delay jitter experienced by each type of media stream. This is sufficient when user has requested only audio or only video. In the case of both audio and video, the delays experienced by both media streams have to be accounted in order to achieve synchronized playback. The client SCM coordinates with the RX agents to exchange delay information and to achieve inter-media synchronization.

3.1.2.1 Design of GUI

The layout of the GUI is shown in figure 3.3. It consists of menus that enable the user to specify the title of the clip, options to select media type as audio only, video only or both audio and video. It has various control panels including media control, volume control and rate control. The session status panel shows the progress of the session with information such as the total length of the clip and the percentage of it that is received.

The user can choose the media type using the options menu. Once the clip is selected using the clip select menu, the media control, volume control and rate control panels are enabled. In the media control panel, the play button is only enabled following clip selection. Once the user selects to play the clip, the other media controls such as pause, rewind, forward and stop are also enabled. If video is one of the media types selected, it is displayed in a separate pop-up window. This video window can be turned off or on using

Figure 3.3: Layout of Graphical User Interface in MPS

toggle video button in the session status panel. Media and volume control panels together provide VCR like controls to the user. In rate control panel, frame rate can be selected.

This is a user QoS setting of the desired frame rate. This is applicable whenever video media type is selected. The user can select a frame rate in a range from 0 to 30 frames/sec.

3.2 Communication Protocols

The communication protocol layers in MPS are as shown in figure 3.4. The MPS application layer sits on top of Session Control Protocol (SCP), audio/video transport and feedback protocol layer. The next lower layer consists of both TCP and UDP, on top of the network layer, Internet Protocol (IP) and lower layers include data link and physical layers. We designed SCP (described in the following section) for controlling a MPS session. A MPS session is an association between a client and a server of our system from the time the client contacts the server to the time it disconnects marking the end of the session. TCP is more appropriate for reliable communication of control messages. We use RTP over UDP for real-time transfer of video/audio data and RTP control protocol, RTCP for QoS feedback. Thus, the communication between the client SCM and server SCM is established using a SCP over TCP/IP; the communication between the media TX agent and its corresponding RX agent is attained using RTP/UDP/IP for data delivery and RTCP/UDP/IP for exchange of control information.

MPS Application		
Session Control Protocol	AV Transport	AV Feedback
	RTP	RTCP
TCP	UDP	
IP		
Link Layer and physical		

Figure 3.4: Communication Protocol Layers in MPS

Three to five logical communication channels are established during a single MPS session.

If the media type requested is only audio then the communication channels are :

- (i) a control channel to control the MPS session
- (ii) an audio data channel
- (iii) an audio feedback channel

If video is also requested, in addition to the above three, there are the following two channels :

- (iv) a video data channel
- (v) a video feedback channel

During initialization, the client goes through a connection establishment phase. In this phase a control channel is opened for communicating with the MPS server(s). The client

sends vital information such as port numbers for video/audio data to the server(s) on this channel. Further details about the protocols are given in the following sections.

3.2.1 Session Control Protocol

We designed SCP for communication between the server SCM and client SCM. It has the following three primary functions :

- (i) to request media specific information
- (ii) to request media delivery
- (iii) to control media delivery

The messages exchanged between the client SCM and server SCM can be categorized as requests, responses and notifications. All three types of messages have the generic message structure which is depicted in figure 3.5. The first three fields form the header of a session control message. The first field in the header indicates the version of the protocol, the second field denotes the operational code to mark the type of the message, the third field is to indicate the total length of the message including the length of both the header and the message body. The last field is optional and is of variable length depending on the type of the message. In this section we refer to server SCM as server and client SCM as client for the sake of brevity.

Figure 3.5: Session Control Protocol Message Structure

The server parses the request messages and passes the necessary information to the TX agents. If the message format is found to be incorrect then an appropriate error code is set and an error message is sent. Similar action is taken by the client. The following are the request messages supported by the SCP:

- *Register* : This message is sent by the client to the server at the time of connection establishment. The message includes fields that contain the audio port number and video port number of audio RX agent and video RX agent respectively. Since RTCP port number is consecutive to RTP port number, this information need not be supplied to the server. The server passes the port number information to the TX agents.

- *Play*: The client signals the server to start sending the media data by sending this message. This message has media type field and a clip title. The clip title is passed to the appropriate TX agent depending on the media type requested.

- *Pause*: Upon receiving *pause* message, the server temporarily halts transmitting media data by informing the appropriate TX agent.
- *Resume*: The server starts sending video/audio data. This message contains offset information that corresponds to a position in the clip in units of milliseconds. The TX agents send media data starting from this position.
- *Stop*: This request signals the server to stop transmitting the video/audio data pertaining to that clip which was last requested using the *play* message.
- *Set-rate*: This request informs the server to transmit video at a specified rate. The information in this message reflects the QoS setting at the client where the QoS metric is the video frame rate. If a frame rate that is more than the present rate at the server is desired, the adjustment is made based on the current network bandwidth availability.
- *End*: The client sends *end* request when it is ready to quit. The server stops any transmission that is in progress to this client.

The following are response messages supported by SCP :

- *Register-resp*: The server sends this message in response to client's *register* message. The information contains the RTCP port numbers of the TX agents. This is useful to provide feedback statistics of reception quality by the RX agents to the TX agents.
- *Vframe-info*, *Aframe-info*: These responses are sent by the server in response to client's *play* request. These give information about the original rate of recording and the length of video, audio respectively of the clip.
- *Rate-resp*: This message is sent to the client in response to *set-rate* request. It indicates the current video frame rate at the server. The current video frame rate at the server is the actual rate at which the server is currently sending out frames after adjusting based on the current network load.
- *Error*: This is typically sent by the server to the client in response to any of the request messages. This message has an error code field that indicates the type of error that has occurred. The error classes and error codes with their description is given in table 3.1.

The notification messages that are used in the SCP are as follows:

- *Vend-notify*, *Aend-notify*: These messages are to notify the end of video and audio media transmission respectively.

Table 3.1: Error Classes and Codes in Session Control Protocol

Error Class	Error Code	Description
	ER_Version	protocol version mismatch
	ER_Opcode	unknown operational code
	ER_MsgLen	mismatch in message length
	ER_Internal	internal system error due to lack of resources
	ER_Undef	undefined error
	ER_MediaType	unknown media type
	ER_ClipName	bad format of clip title
	ER_Open_Cfile	control file open failure
	ER_Open_Vfile	video file open failure
	ER_Open_Afile	audio file open failure
	ER_Read_Cfile	control file read failure
	ER_Read_Vfile	video file read failure
	ER_Read_Afile	audio file read failure
	ER_Access	access permission violation
	ER_Vformat	unsupported video format
	ER_Aformat	unsupported audio format

All types of messages in session control protocol of MPS are summarized in table 3.2.

The columns indicate operational code of the message, the field in the message body, the result or error sent back as a response to this message using session control protocol.

Note that the result of *play* message is media data arriving at the RX agents' data ports using RTP. The table shows only those results that are sent using session control protocol.

Table 3.2: Messages in Session Control Protocol

Operational Code	Message field(s)	Result	Errors
OP_Register	audio RX RTP port video RX RTP port	audio TX RTCP port video TX RTCP port	ER_Generic
OP_Play	media type clip title	-none-	ER_Media ER_File ER_Format ER_Generic
OP_Pause	-none-	-none-	ER_Generic
OP_Resume	offset	-none-	ER_File ER_Generic
OP_Stop	-none-	-none-	ER_Generic
OP_Set-rate	frame rate	adjusted frame rate	ER_Generic
OP_End	-none-	-none-	ER_Generic
OP_Register-resp	audio TX RTCP port video TX RTCP port	-none-	ER_Generic
OP_Error	error code	-none-	ER_Generic
OP_Vframe-info	record rate video length	-none-	ER_Generic
OP_Aframe-info	record rate audio length	-none-	ER_Generic
OP_Rate-resp	adjusted frame rate	-none-	ER_Generic
OP_Vend-notify	-none-	-none-	ER_Generic
OP_Aend-notify	-none-	-none-	ER_Generic

During an MPS session the server is in one of the following states:

- **IDLE**: the initial state of the server. In this state the server waits for a new client *register* request, after which it transits to **WAIT** state.
- **WAIT**: the server waits for additional requests from the client.
- **PLAY**: an active transmission state of the server. The server enters this state when it receives a valid *play* or *resume* request.
- **STOP**: the server enters this state from **PLAY** state when transmission is halted.

Figure 3.6: MPS Server State Transition Diagram

The server can go to the IDLE state from any other state when it receives an *end* request from the client. The server state transition diagram is shown in figure 3.6. The server state reflects the state of the client and thus the client need not remember its own state as it is always monitored by the server.

3.2.2 Media Transport and RTP

In our system RTP is used for audio and video transport. The following sections give the details of RTP followed by audio and video packetization using RTP.

3.2.2.1 Real-time Transport Protocol (RTP)

The roots of RTP date back to the early versions of the protocol used in Visual Audio Tool, *vat*, [JAC92] a packet audio application which later came to be known as RTP version 0. In December 1992 RTP version 1 was published by Henning Schulzrinne, GMD Berlin which went through several states of Internet drafts. The current RTP is version 2 which is an effort of Audio-Video-Transport (AVT) working group of the Internet Engineering Task Force (IETF) [SCH96a] that got approved on November 2, 1995 by the Internet Engineering Steering Group (IESG) as a proposed standard. The latest extensions have been made by an industry consortium who use RTP as the basis of their Real Time Streaming Protocol (RTSP) [RAO96].

RTP has been specified by the AVT group of IETF as an end-to-end application level protocol for the transport of real-time multimedia data. RTP does not ensure in-time delivery of data. This requires the support of lower layers that actually have control over resources in switches and routers as in RSVP [ZHA93]. The intention of the word real-time in the context of the protocol is not to receive the streams the same time as they were sent. But it stands for an attribute of stream based transmission, for which the original

framing and timing of the data chunks often referred to as packets can be reproduced after transmission over the network by the recipient. The main goals of RTP include :

- transmission of media streams over any network connection.
- support different payload types.
- provide information needed to synchronize various streams.
- flow and congestion control information.

Figure 3.7: RTP and the Protocol Stack

RTP has two parts : the data delivery protocol, and the control protocol, RTCP. The data delivery protocol handles the actual media transport, while RTCP manages control information. The main functions of RTP are to provide encoding, timing and sequence numbering information for real-time data. It is independent of lower protocol layers and hence can be used over UDP/IP or directly over ATM adaptation layer (figure 3.7).

Typically, for real-time delivery of audio and video in an internetwork, a connection-less transport protocol such as UDP is preferred over reliable connection-oriented transport protocol such as TCP. The main reasons for reliable transport protocols such as TCP to be inappropriate for the delivery of real-time data such as audio and video are as follows:

- Reliable transmission as in TCP increases the delay by at least one round-trip time or even more due to re-transmissions. This results in increased delay in reconstruction, creating an audible or noticeable gap in the presentation of audio or video.
- The congestion control method in TCP is based on a slow start algorithm which decreases the congestion window when packet losses are detected. This results in starvation at the receiver in the case of real-time data.
- The TCP header size is 40 bytes which is much bigger compared to the 8 bytes of UDP header size.

Both TCP and UDP lack the encoding and timing information as provided by RTP which are necessary to decode and synchronize the media streams at the receiver.

RTP follows the principles of Application Level Framing (ALF) [CLA90], which does not overly restrict the application design, instead, gives flexibility to the application to respond to situations of congestion and packet loss. In adherence to its ALF philosophy, some fields in the RTP header are deferred to an “RTP Profile” document which defines the semantics according to the given application, an example is RTP profile document for audio and video conferences [SCH96b]. The interpretation of fields can be further refined by the “Payload Format Specification”. An example payload format specification is RTP/JPEG specification [FEN96] for motion-JPEG video. We briefly mention some fields in RTP header. Complete details on RTP specification can be found in [SCH96a]. Figure 3.8 shows the RTP fixed header.

Figure 3.8: RTP Fixed Header

- *marker bit* : its interpretation depends on the *payload type* indicated by PT. If M is set, it marks the end of a frame for video, start of a talk spurt for audio. When there is no silence suppression for audio it remains zero.
- *payload type*: indicates the type of data that is carried. For instance payload type can be JPEG video or PCM audio. A summary of valid payload types is given in table 3.3. It is to be noted that H.323 [ITU96] is not yet supported as a valid audio payload type.
- *Sequence number*: this is a number that increases by one for each RTP packet sent. It allows loss detection and reordering of RTP packets at the receiver.
- *Media timestamp*: indicates the frequency of timestamp which varies depending on payload type. For example 90 KHz for JPEG video and 8 KHz for PCM audio are recommended [SCH96b]. It increases monotonically for every audio packet sent or every video frame that is sent. All packets that belong to one video frame carry the same timestamp.
- *Synchronization Source id*: uniquely identifies the source of the audio or video data.

Table 3.3: Payload Types for Standard Audio and Video Encodings*

PT	Encoding Name	audio/video (A/V)	clock rate (Hz)
0	PCMU	A	8000
1	1016 (celp)	A	8000
2	G721	A	8000
3	GSM	A	8000
4	unassigned	A	-
5	DVI4	A	8000
6	DVI4	A	16000
7	LPC	A	8000
8	PCMA	A	8000
9	G722	A	8000
10	L16	A	44100
11	L16	A	44100
12	unassigned	A	-
13	unassigned	A	-
14	MPA	A	90000
15	G728	A	8000
16-23	unassigned	A	-
24	unassigned	V	-
25	CelB	V	90000
26	JPEG	V	90000
27	unassigned	V	-
28	nv	V	90000
29	unassigned	V	-
30	unassigned	V	-
31	H261	V	90000
32	MPV	V	90000
33	MP2T	AV	90000
34-71	unassigned	?	-
72-76	reserved	N/A	N/A
77-95	unassigned	?	-
96-127	dynamic	?	-

*Data from [SCH96b].

3.2.2.2 Video Transport

To transmit JPEG video using RTP the AVT working group of IETF has specified RTP payload format for JPEG-compressed video [FEN96]. After the RTP header, JPEG header (figure 3.9) is appended which is then followed by the actual media data. The JPEG header and the actual media data comprise RTP payload. Some fields in JPEG header are discussed below.

Figure 3.9: JPEG Header

- *Fragment Offset*: it indicates data offset of the current packet in the JPEG frame in bytes. It is useful to reconstruct a frame by the receiver.
- *Type*: indicates the type of JPEG decimation. Two types are currently valid. Type 0 is used for 4:2:2 decimation and type 1 for 4:1:1 decimated JPEG.
- *Q*: reflects the quality factor. *Type* and *Q* fields together are used by the receiver to deduce the selection of Huffman and quantization tables to be used to decode the JPEG frame. Typically the value of *Q* ranges from 1 to 99.

Video transport in MPS is undertaken by the video TX agent. We support motion-JPEG for video. Motion-JPEG is essentially a series of still images, each compressed independently using JPEG compression standard [PEN93]. Although there is no standard file format for motion-JPEG at present, both vendor specific hardware and software are available for encoding and decoding motion-JPEG video. As each frame can be decoded independent of other frames, it is often a popular choice for video transmission over unreliable packet networks.

Figure 3.10: Video Packetization in MPS

The video packetization procedure in MPS is depicted in figure 3.10. Video TX agent relies on m-JPEG control file (MCF) to attain random access to a m-JPEG file without requiring sequential search. MCF file is created using a separate parse utility at the time a m-JPEG file is stored in AV data-store to contain control information pertaining to that

particular m-JPEG file. The information contained in MCF file includes the total number of frames, frame rate at which the video was recorded, frame offsets and the name of the m-JPEG file to which the control information corresponds. Video TX agent reads MCF file and m-JPEG file from AV data-store that correspond to the requested clip. MCF file to locate a frame in the m-JPEG file.

Since JPEG frames are often huge, video TX agent fragments each frame to smaller portions of frame data. For video, RTP packet is formed by adding RTP fixed header, JPEG header followed by a portion of frame data. The typical size of an RTP packet in MPS is 1024 bytes. All RTP packets that belong to a single JPEG frame carry the same media timestamp. The marker bit is set on the last RTP packet of a frame. The fragment offset field is set to the byte offset of this packet as in the original frame. RTP packets in MPS are sent to the UDP port of video RX agent .

Video RX agent reorders the RTP packets as they arrive at the designated UDP port based on sequence number information. An entire frame is reconstructed as a sequence of RTP packets beginning with a packet having a zero fragment offset and ending with a packet having RTP marker bit set to one.

3.2.2.3 Audio Transport

Audio transport in our system is done by audio TX agent. We support PCMU (μ -law) payload type for audio. Audio TX agent reads PCM audio file from the AV data-store that corresponds to the clip specified by the client. It locates the data offset from the header information contained in the file, reads PCM audio data starting from that data offset and packetizes into RTP packets as shown in figure 3.11. In this case, RTP header is followed by PCM audio data, no extra payload specific header is required as in the case of video. The marker bit in RTP header is set to zero in all RTP audio packets.

Figure 3.11: Audio Packetization in MPS

A tradeoff exists between the audio packet size and the associated packet overhead. Smaller audio packets increase header overhead and on the other hand large audio packets make gaps more noticeable during playback when packets are lost. Audio packet size has to be set to an optimal value in order to account for poor aural quality during packet loss as well as high header overhead and to obtain a balance between the two. In MPS an audio RTP packet typically contains 40 ms of audio data. An occasional loss of audio packets of this size is tolerable and does not degrade the aural presentation quality at the receiver.

Audio RX agent reorders the RTP packets as they arrive at the designated UDP port based on sequence number information contained in the RTP header.

3.2.3 Media Feedback and RTCP

We use RTCP which is a part of RTP specification [SCH96a] for providing feedback information on the reception quality of audio and video media. RTCP and media feedback are described in the following sections.

3.2.3.1 Real-Time Control Protocol (RTCP)

RTP has a control protocol called RTCP that supports the following main functions:

- QoS monitoring and congestion control.
- Inter-media synchronization.

The other functions are sender identification, session size estimation and scaling. RTCP messages are exchanged by the source and destination of a media stream on a per stream basis. An RTCP message consists of a number of stackable packets, each with its own type code and length indication. Several RTCP packet types are defined to carry a variety of control information. We describe two packet types, the Sender Report (SR) and Receiver Report (RR) packets that are relevant for our discussion. Complete details on RTCP can be found in the RTP specification [SCH96a]. The RTCP header is shown in figure 3.12.

Figure 3.12: RTCP Header

The SR or RR information follows the RTCP header. The SR is issued by the sender to indicate the transmission statistics. The fields in SR that follow the header are shown in figure 3.13. The first 2 bytes in SR indicate the Network Time Protocol (NTP) [MIL92] timestamp and the next byte indicates the RTP timestamp. The NTP timestamp indicates the wall clock time at the sender when the SR was generated, this can be used to estimate round-trip delays.

Figure 3.13: RTCP Sender Report (SR) Information

The RR is sent by the receiver to indicate the reception statistics. RR packet information is shown in figure 3.14. The first byte of RR indicates the source identifier of the media transmitter to which the remaining information pertains. The fraction lost gives the

Figure 3.14: RTCP Receiver Report (RR) Information

fraction of RTP packets that were lost since the last report. The sender can sense the network state based on the estimates of inter-arrival jitter and packet loss.

3.2.3.2 Media Feedback

RTCP is used in MPS to provide feedback on media reception. A UDP port number consecutive to the RTP port is used by each TX agent to send and receive RTCP packets. Similarly each RX agent at the client uses a UDP port that is one greater than the RTP port number to exchange RTCP messages. The TX and RX agents exchange control messages on a periodical basis. The bandwidth used for RTCP messages is recommended to be 5% of the bandwidth used for RTP data packets.

In MPS, audio and video TX agents generate RTCP SR and send to the audio and video RX agent respectively. RX agents generate RTCP RR. By sending the RR report the RX agents provide feedback on quality of reception. This feedback is used by the TX agents to control the QoS by dynamically adjusting the delivery rate.

Chapter 4

Synchronization in a Distributed Environment

4.1 Introduction

Synchronization is required for end-to-end temporal presentation of multimedia information. As noted in chapter 2, there are two aspects to the problem of multimedia synchronization. They are synchronization definition and synchronized delivery [STE92]. Synchronization definition deals with the specification of the spatial and temporal relationships among the media objects. The basic temporal relations according to [ALL83] are shown in figure 4.1. Inverse relations are not listed in the figure. These can be broadly classified as sequential and simultaneous relationships. Besides synchronization definition mechanisms we also need delivery mechanisms in order to satisfy the temporal relations during presentation.

Synchronized delivery in a distributed multimedia system involves three levels: source, transport and destination level. Synchronization at the server or source deals with the retrieval of media from the computer's persistent storage or live from a media device. The

source should retrieve media at a rate that matches the original rate of recording or at a rate requested by a client which can be remote or local to the server. Transport level synchronization deals with bandwidth allocation and QoS guarantees over a communication network. This can be realized through protocols that have control over the resources in the routers and switches such as RSVP [ZHA93]. The client or the destination is responsible for reassembly and synchronous playback of media. All three levels need to work in conjunction to provide a reliable multimedia information system which will deliver the media according to their definitions.

Figure 4.1: Typical Temporal Relationships

Multimedia synchronization systems must be able to handle both intra-media and inter-media synchronization definitions. Intra-media synchronization refers to the time relation between various segments belonging to one continuous media object. An example is the time relation between the single frames of a video sequence. For a video with a rate of 25

frames per second, each frame must be displayed every 1/25th second or every 40 ms. For 8000 samples per second audio, each sample must be played every 1/8000th second. The intra-media synchronization mechanism should ensure the continuity by rendering the media segments at the defined rate.

Figure 4.2: Data Location Models for Distributed Systems

Inter-media synchronization refers to the synchronization between media objects. This type of synchronization is to meet the temporal relations such as sequential or simultaneous. If for example simultaneous audio and video is required, in addition to maintaining the continuity within a single media object, the rate of both media objects should be matched so that the audio and video correspond to each other.

Both the sources and destinations of the media are responsible for maintaining synchronization. In distributed systems, there are four generalized data location models [BUL91]. These models shown in figure 4.2 are the local single source, local multiple sources, distributed single source and distributed multiple sources. Because of the different possible topologies between sources and destinations which may result in varying data paths for distributed media, most distributed multimedia information systems rely mainly on the destinations to maintain synchronization [EHL94].

Table 4.1: Quality of Service for Synchronization Purposes*

Media		Mode, Application	QoS
video	animation	correlated	+/- 120 ms
	audio	lip synchronization	+/- 80 ms
	image	overlay	+/- 240 ms
		non-overlay	+/- 500 ms
	text	overlay	+/- 240 ms
		non-overlay	+/- 500 ms
audio	animation	event correlation (e.g. dancing)	+/- 80 ms
	audio	tightly coupled (stereo)	+/- 11 us
		loosely coupled dialogue mode with various participants)	+/- 120 ms
	image	tightly coupled (e.g. music with notes)	+/- 5 ms
		loosely coupled (e.g. slide show)	+/- 500 ms
	text	text annotation	+/- 240 ms
	pointer	audio related to the item to which the pointer shown	- 500 ms + 750 ms

*Data from [STE93].

In a distributed multimedia information system there are many sources that contribute to the complexity of the synchronized delivery of multimedia information. In the next section, we present a review of existing synchronization mechanisms. In section 4.3 a

synchronization technique that is suitable for use in MPS is discussed. We address synchronization undertaken by the receiver that meets the simultaneous temporal relationship between audio and video at a low level or raw media level compared to those discussed in section 4.2. Human perception of synchronization allows for a certain skew between media as shown in table 4.1. We take this QoS for synchronization into account when achieving inter-media synchronization. Specific methods on how to achieve synchronization using the information in RTP packets are described.

4.2 Synchronization Mechanisms

Synchronization is one of the major issues in developing distributed multimedia applications. The time constraints within and among multiple media streams imposed during composition need to be maintained during playback in order to achieve synchronized presentations. For these reasons synchronization has been recognized as one of the central problems in multimedia system development [LIT90, STE90].

Substantial research has been reported on the topic of synchronization in multimedia applications. In [STE92], Steinmetz introduced the concept of Logical Data Unit (LDU). A continuous media stream is viewed as a sequence of LDUs. Synchronization control is performed on a LDU. It requires a connection-oriented data transport system in order to guarantee the simultaneous data delivery on multiple channels. In order to overcome the problem of rate matching among various media streams, a composite LDU stream is

created to be transferred on a single connection. Transmitting composite media on a single connection has several drawbacks as noted in earlier chapters.

Ferrari [FER92] proposes a delay jitter control scheme that ensures media synchronization under the conditions that delay bound is guaranteed and source and destination clocks are in synchrony. For end-to-end delay to be guaranteed every node in the transmission path from the source to the destination is involved in synchronization control by regulating the random delay that an arriving packet experiences from the previous hop. This mechanism adds extra workload on intermediate nodes and its deployment is complex in reality. In [ESC92], Escobar et al. address flow synchronization control protocol by implementing an end-to-end synchronization delay. The random delay experienced by each packet is regulated at the destination node. The actual delay experienced by a packet cannot exceed the preset end-to-end delay for the network connection and a global accurate clock is required.

Little and Ghafoor [LIT90] establishes a synchronization specification and storage model in which a logical database structure constructed from *object composition Petri net (OCPN)* can be used to specify and store synchronization and composition of multimedia entities. In [LIT91], Little et al. propose computing schedules before transmitting the data packets. The multimedia traffic runs according to the schedule to reach the destination in time for presentation. However, if the underlying network introduces random delays, scheduling and predicting the traffic is not enough to maintain simultaneous data delivery. Delay compensation mechanism at the receiver is necessary. Woo [WOO94] describes a

synchronization framework for preorchestrated information using *extended object composition Petri net (XOCPN)*. XOCPNs are used by both transmitter and receiver to specify the time constraints among multiple media streams. Delivery of data is prescheduled at the source based on *a priori* information on network delay jitter characteristics.

In [RAN92] Rangan et al. present an inter-media synchronization technique for multimedia on-demand services. The centralized multimedia server uses feedback units transmitted by the receivers to detect asynchronies among the media streams. Resynchronization is performed at the server by speeding up or slowing down the traffic on various media streams. With this scheme, under heavy network load condition it might be difficult to detect and correct the asynchronies before they are noticed by the user. In [LI92], Li et al. present compensation mechanisms at the receiver to recover synchronization errors that occur due to random network delays.

Nicolaou [NIC90] describes a two-level synchronization scheme in which the lower level can be considered as an ordered sequence of samples such as video frames, referred to as *physical synchronization frames (PSF)*, and the higher level is structured as an ordered sequence of *logical synchronization frames (LSF)* which consists of a number of PSFs and can be specified by applications. The correspondence between PSF and LSF allows application to specify and monitor synchronization at LSF level, whereas these synchronizations are performed by the communication subsystem at PSF level. In this way, both fine-grained and coarse-grained synchronization mechanisms are provided. The LSF

is somewhat related to *data framing* concept in [RAV93] as both LSFs and data frames form the logical units of synchronization in a level above the raw data streams. Ravindran et al. [RAV93] proposes various compensation mechanisms based on network delay models with the use of a global clock. Shivakumar et al. [SHI95] presents an algorithm that allows dynamic update of buffer size based on estimated network delay in order to achieve application-specified acceptable delay and loss rate. Synchronization among multiple data streams is realized implicitly by satisfying the stream with the most stringent delay requirement.

Typically, research on synchronization issues has been in a general context. A survey on media synchronization [BLA96] shows that a large number of synchronization supporting systems have been developed. Although many prototypes and products have implemented some kind of synchronization, it is still an open research issue to find out the most appropriate approaches for performing synchronization, especially in distributed environments.

4.3 Media Synchronization in MPS

A major factor in achieving synchronization in a distributed environment is the end-to-end delay estimation. The requirements to achieve intra-media synchronization are end-to-end delay measurement and re-assembly buffering to smooth the variable delay. To achieve inter-media synchronization additional requirements are common playout time estimation

and negotiation facilities between the media receivers. The source sends out packets at a constant rate and under ideal conditions, after the first arrival the destination receives the rest at the same rate as they were sent. However in reality the arrival rate at the receiver may not be constant due to a variety of delays that comprise end-to-end delay (figure 4.3). The different delays that attribute to end-to-end delay estimation are: (i) data retrieval delay, attributed to disk access delays in the case of stored media, (ii) packetization delay, (iii) network delay, this includes propagation and queuing delays, (iv) depacketization delay, and (v) decompression and rendering delays. Under the assumption that the media data blocks are physically located in the local disk space of the server and light

Figure 4.3: End-to-End Delay Variation at the Receiver

workstation load conditions the first two can be assumed to vary only slightly. The last two delays are also estimable under light workstation load [KOU96]. However the main factor of concern in end-to-end delay estimation is network delay which varies randomly depending on network traffic load.

The difference between the end-to-end delay of audio and video also vary considerably. Storing and sending both audio and video in an interleaved format simplifies synchronization during playback as both media experience the same delay. However, interleaving makes adaptation difficult. In our system audio and video are separately stored as different files and are sent as separate streams making adaptation practical. Adaptation is achieved by dropping video frames at the source thus making degradation graceful. Audio on the other hand is never dropped as it results in undesirable annoyance to the end user.

Figure 4.4: Association Between Audio and Video Frames

In our approach to synchronization grouping relationships or association between individual video frames and audio packets is maintained to synchronize video to audio. Figure 4.4 illustrates the association between audio and video that correspond to a clip of one second duration, with a video frame rate of 20 frames per second; each audio frame contains 0.2 seconds of audio; every 4 consecutive video frames correspond to every 0.2

seconds of audio. Synchronized playback can be realized when corresponding audio and video frames are played back simultaneously. Media synchronization between audio and video can be perceived with a skew of 80-100 ms which is below the level of human perception of synchronization [STE93, KOU96].

The methodology for intra-media synchronization at the receiver is: i) to estimate smoothed inter-arrival jitter as each packet arrives; ii) estimate the presentation time of the first unit (video frame or audio packet) by offsetting the actual arrival time to an additional amount so as to accommodate later arrivals; iii) given a maximum allowable jitter that can be tolerated, estimate the presentation times of subsequent units such that the actual jitter experienced does not violate the jitter bound. For inter-media synchronization, above mentioned steps i and ii are followed, step iii) schedule presentation time of first unit such that both media units start at the same time; iv) at every synchronization timer event, estimate a common range of allowable jitter based on current jitter experienced by both media units such that audio jitter bound is not violated; v) presentation of audio and video follow the schedule allowing maximum skew that can be tolerated while presenting video.

In MPS, RX agents use media timestamps contained in the RTP header of video/audio packets to smooth out the effects of network delay jitter. An estimate of inter-arrival jitter can be calculated based on the timestamp information in media timestamp units. Inter-arrival jitter is the mean deviation of the difference in packet spacing at the receiver compared to the sender for a pair of consecutive packets [SCH96a]. Audio RX agent calculates smoothed inter-packet-arrival jitter estimate for every audio packet it receives

while video RX agent estimates smoothed inter-frame-arrival jitter for every video frame that is received in seconds as shown in figures 4.5 and 4.6 respectively. Note that the video RX agent updates jitter estimate for every frame when there is a change in RTP timestamp by checking the timestamp in each packet. This follows logically from the fact that all packets that belong to a single video frame bear the same timestamp.

<p>if packet = 1st packet</p> <p>$T_a[0]$ = RTP timestamp</p> <p>$A_a[0]$ = Present system time</p> <p>$J_a = J_{a\ rtp} = 0$</p> <p>else</p> <p>$T_a[i]$ = RTP timestamp</p> <p>$A_a[i]$ = Present system time</p> <p>$D_{nd(i,i-1)} = (A_a[i] - A_a[i-1]) * C_{af} - (T_a[i] - T_a[i-1])$</p> <p>$T_a[i-1] = T_a[i] ; A_a[i-1] = A_a[i]$</p> <p>$J_{a\ rtp} = J_{a\ rtp} + (D_{nd(i,i-1)} - J_{a\ rtp}) / 16$</p> <p>$J_a = J_{a\ rtp} / C_{af}$</p>	<p>$T_a[i]$ - RTP timestamp of ith audio packet</p> <p>C_{af} - Audio clock frequency in Hz</p> <p>$A_a[i]$ - System time when ith packet arrived</p> <p>D_{nd} - Difference in relative transit delay between two consecutive packets</p> <p>$J_{a\ rtp}$ - Inter-arrival jitter for audio packets in RTP timestamp units</p> <p>J_a - Inter-arrival jitter for audio packets in sec</p>
---	---

Figure 4.5: Estimation of Inter-Arrival Jitter in Audio Packets

At this point we clarify that RTP timestamps are not absolute but are relative timestamps. The audio TX agent attaches a timestamp for the first audio packet that can start from zero or a random value. For each subsequent audio packet the timestamp is incremented by a factor of C_{af}/R_{ap} , where C_{af} is the audio clock frequency and R_{ap} represents audio packet rate. For instance, PCM μ -law audio has C_{af} of 8000 Hz and R_{ap} varies based on the number of samples packed into each audio packet for transmission. If we pack 320 8-bit samples of audio to form a 40 ms packet, then R_{ap} can be deduced to be 25 as we need

<i>if packet = 1st packet</i> $T_v[0] = \text{RTP timestamp}$ $A_v[0] = \text{Present system time}$ $J_v = J_{v\ np} = 0$ <i>else if change in RTP timestamp</i> $T_v[i] = \text{RTP timestamp}$ $A_v[i] = \text{Present system time}$ $D_{rd(i,i-1)} = (A_v[i] - A_v[i-1]) * C_{vf} - (T_v[i] - T_v[i-1])$ $T_v[i-1] = T_v[i] ; A_v[i-1] = A_v[i]$ $J_{v\ np} = J_{v\ np} + (D_{rd(i,i-1)} - J_{v\ np}) / 16$ $J_v = J_{v\ np} / C_{vf}$	$T_v[i]$ - RTP timestamp of <i>i</i> th video frame C_{vf} - Video clock frequency in Hz $A_v[i]$ - System time when 1st packet of <i>i</i> th video frame arrived D_{rd} - Difference in relative transit delay between two consecutive frames $J_{v\ np}$ - Inter-arrival jitter for video frames in RTP timestamp units J_v - Inter-arrival jitter for video frames in sec
---	--

Figure 4.6: Estimation of Inter-Arrival Jitter in Video Frames

to send 8000 bytes of audio every second. In essence, for a 40 ms audio packet, each carries a timestamp that is incremented by 320.

Video TX agent timestamps all packets that belong to the first video frame with a value of zero or a random number and from then on increments the timestamp for each video frame by a factor of C_{vf}/R_{vf} , where C_{vf} is the video clock frequency and R_{vf} represents original rate of video frame recording. A formula to compute the timestamp for arbitrary frame number and audio packet number is shown in figure 4.7. Since the audio and video are pre-recorded we can calculate the total video frames and the total audio packets from the length of audio and video and the record rate or audio packet size. Thus, each TX agent can compute RTP timestamps in the case of pre-recorded media before transmission as in the case of live media by using the three factors, the current audio packet or video frame number, clock frequency and the recording rate.

<p>a) Audio:</p> $T_{ak} = (k-1) * C_{af} / R_{ap} + O_a, \quad k = \{1, N_a\}$	<p>T_{ak} - RTP timestamp of kth audio packet C_{af} - Audio Clock frequency in Hz R_{ap} - Rate of audio packets O_a - Random Offset N_a - Total Number of audio Packets</p>
<p>b) Video:</p> $T_{vj} = (j-1) * C_{vf} / R_{vf} + O_v, \quad j = \{1, N_f\}$	<p>T_{vj} - RTP timestamp of jth video frame C_{vf} - Video Clock frequency in Hz R_{vf} - Rate of video frames as recorded O_v - Random Offset N_f - Total Number of video frames</p>

Figure 4.7: Audio and Video RTP Timestamp Generation

<p>Audio Only:</p> <p>1. if (packet = 1st packet) $P_a[0] = A_a[0] + O_{id} + D_{ad}$ else $I_a[i] = (T_a[i] - T_a[0]) / C_{af}$ $J_{ac} = \min \{J_{amax}, J_a\}$ $P_a[i] = P_a[0] + I_a[i] + J_{ac}$</p> <p>2. if ($P_a[i] < A_a[i]$) then - Discard, Play silence if ($P_a[i] > A_a[i]$) then - wait for ($P_a[i] - A_a[i]$) if ($P_a[i] = A_a[i]$) then - Play</p>	<p>$P_a[i]$ - Playout time of ith audio packet O_{id} - Initial playout offset $I_a[i]$ - Time index of ith audio packet D_{ad} - Audio device delay J_{amax} - Maximum jitter allowed for audio J_{ac} - Jitter compensation for audio</p>
---	--

Figure 4.8: Synchronization Mechanism for the Case of Audio Only

The RX agents use these timestamps to serve as an index to position each frame or audio packet in accordance to their original timing relationships allowing for intra-media synchronization. A circular buffer similar to that used in *network voice terminal (NEVOT)* [SCH92] may be used to accommodate early arrivals, and to play out the packets or frames sequentially. As the packets or frames arrive they take position in the buffer based on the timestamp information and can be played out from the head of the buffer.

A playout schedule can be generated as each audio packet is received and its playout can be triggered as per schedule. For the case of audio only, the synchronization mechanism is as shown in figure 4.8. In the case of video only, a similar but less stringent schedule can be allowed by setting J_{vmax} to a very large value to the order of seconds, as missing a deadline will lead to a slower display rate and it is not annoying to the user. This is shown in figure 4.9. The playout of the first audio packet or video frame is offset by O_{id} . The setting of O_{id} is an engineering decision to allow for subsequent packets to arrive in time and to minimize their rejection as late arrivals. The subsequent playout times are scheduled with an addition of $I_a[i]$ or $I_v[i]$ that accounts for original spacing and jitter compensation J_{ac} or J_{vc} . A maximum allowable bound on jitter can be regulated by setting J_{ac} or J_{vc} to a minimum of the current jitter estimate and a preset maximum jitter value J_{amax} or J_{vmax} that can be allowed.

<p><i>Video Only:</i></p> <p>1. if (packet = 1st packet) $P_v[0] = A_v[0] + O_{id} + D_{vd}$ else if (change in RTP timestamp) $I_v[i] = (T_v[i] - T_v[0]) / C_{vf}$ $J_{vc} = \min \{J_{vmax}, J_v\}$ $P_v[i] = P_v[0] + I_v[i] + J_{vc}$</p> <p>2. if ($P_v[i] < A_v[i]$) then - Discard, Play last frame if ($P_v[i] > A_v[i]$) then - wait for ($P_v[i] - A_v[i]$) if ($P_v[i] = A_v[i]$) then - Play</p>	<p>$P_v[i]$ - Playout time of ith video frame O_{id} - Initial playout offset $I_v[i]$ - Time index of ith video frame D_{vd} - Video decode & render delay J_{vmax} - Maximum jitter allowed for video J_{vc} - Jitter compensation for video</p>
--	---

Figure 4.9: Synchronization Mechanism for the Case of Video Only

In the case of audio and video, where inter-media synchronization is needed we need to estimate a common playout point so that corresponding audio and video is played out simultaneously. The exchange of synchronization information and negotiation of playout time between RX agents is coordinated by the client SCM. The synchronization mechanism for this case is shown in figure 4.10.

The initial common playout time $P_c[0]$ is selected to be the maximum of $P_a[0]$ and $P_v[0]$. The RX agents inform the jitter experienced J_{ac} , J_{vc} on a periodic basis whenever the synchronization timer event occurs which can be set by the application to achieve coarse-grained or fine-grained synchronization. Figure 4.4 illustrates an example where synchronization timer event occurs every 0.2 sec. The SCM gives a common range of allowable jitter to be applied by the RX agents when scheduling the presentation times for the subsequent audio packets or video frames. Video RX agent can allow a maximum skew, S_{max} that can be tolerated during presentation. The RX agents follow the playout schedule until the new range of allowable jitter is informed by the SCM. If the actual arrival time is greater than the scheduled playout time the frame is discarded. If the frame has arrived early, the playout is delayed until its actual playout time. If the arrival is on time it is decoded and displayed. The same is applicable for audio. This allows for inter-media synchronization.

Figure 4.10: Synchronization Mechanism for the Case of Audio and Video

The RTP and RTCP packets contain necessary information to carry out synchronization operations in complex scenarios such as a conferencing application where several live media streams that belong to each participant have to be identified and synchronized. For this purpose every participant has a unique CNAME (canonical name). Each Media stream that belongs to a single participant has a unique source id but a common CNAME, thus enabling to group audio and video of a participant. The timestamp in RTP header is media specific and different reference clocks are used for audio and video. In order to estimate playout offset or delay that is artificially added to audio or video for common playback, a common reference clock is required. The RTCP SR packet contains NTP (Network Time

Protocol) [MIL92] timestamp along with a reference to the media timestamp contained in RTP packet. The NTP timestamp which gives the wall clock time at the sender provides a common reference clock. This can be used to associate corresponding audio and video at the receiver. In MPS, since there is one audio and video stream to be associated we do not require the CNAME information and on the other hand since the original rate of recording is notified to client through *Vframe-info*, *Aframe-info* response messages, the NTP timestamp information in the SR report is not crucial in order to achieve inter-media synchronization.

4.4 Summary

In this chapter we gave an overview of several synchronization terms that are widely used including inter-media and intra-media synchronization. We also revised the basic temporal relationships that exist among the media objects. A survey of several synchronization algorithms that deal with the problem of synchronization specification and delivery is presented. These algorithms perform synchronization at different levels and in a variety of frameworks. Some are receiver-based and others are source-based. Some account for the end-to-end delay by compensating it with a fixed metric. Adaptive delay compensation algorithms better suit packet networks that induce random delays.

We presented synchronization mechanisms for use in MPS. This mechanism is receiver-based and is tailored to meet the simultaneous temporal relationship between audio and video media objects. We described inter-media synchronization between audio and video

keeping audio as the main stream and synchronizing video to audio using synchronization timer events. We have shown how synchronization can be achieved for all the three cases that MPS presently deals with, audio only, video only, audio and video. These are based on the RTP timestamp information contained in audio and video packets. We have shown how inter-arrival jitter can be estimated using this information. We have also illustrated delay compensation and adaptive playout mechanisms based on the jitter estimates.

Chapter 5

Quality of Service Control

5.1 Introduction

A fundamental characteristic of any communication system is the Quality of Service (QoS). The QoS is defined as the collective effect of service performance which determines the degree of satisfaction of a user of the service [CCI88]. Since different applications have different requirements, services are represented using certain parameters. Service parameterization is defined in ISO (International Standard Organization) standards through the notion of Quality of Service (QoS).

Traditional QoS (as in ISO standards) referred to measures at the network layer of the communication system. However, for distributed multimedia systems, the notion of QoS needs further extension into several QoS layers [STE95]. In general, the application, system and device layers are present in all distributed multimedia systems (figure 5.1). The user layer exists on top of these layers if a human user is also present such as at the client.

These QoS layers broadly correspond to three levels of QoS as perceptual, application or presentation and resource level QoS.

Figure 5.1: QoS-Layered Model for Multimedia Communicating System

User-level QoS parameters specify the service quality that the user sees or hears. This is hard to quantify and there are several issues that need to be addressed such as user perception and user preferences. The user may not perceive a single missing frame as a reduction of quality. The user preferences may be task-dependent. For example, in an educational video, the user may want to watch all frames even at a lower display rate and in a sports video the user may prefer a high frame rate tolerating a few frame drops. Presentation-level QoS parameters describe requirements for application services that may be specified in terms of media quality and media relations. Some QoS parameters for the presentation of a media object are summarized in table 5.1. Parameters such as decoder buffer size, bandwidth, end-to-end delay can also be used to specify QoS at this level. In

contrast to user-level, QoS at this level can be described easily. Resource-level QoS parameters define quality in terms of resources for particular devices, for instance CPU cycles; disk bandwidth, resolution and colors of a display device; network throughput, packet size, packet loss rate, end-to-end delay, packet priority etc. Parameters at this level have a significant influence on over-all system performance.

Table 5.1: Quality of Service for Media Object Presentation

Quality of Service	Color Depth	Color Depth	Lin. or log sampling
	Resolution	Resolution	Sample Size
		Frame Rate	Sample Rate
		Jitter	Jitter
		Error Rate	Error Rate

Based on the level of performance guarantees that they provide, the approaches to resource management can be categorized as guaranteed, predictive, and best effort. Guaranteed services provide QoS guarantees as specified through QoS parameter values or range of values in either deterministic or statistical representation. Predictive approach is based on past network behavior and hence QoS parameters are estimates of past behavior which the service tries to match. Best-effort service is based on either no guarantees or no partial guarantees. There is either no specification of QoS parameters required, or some bounds in deterministic or statistical forms are given. Most current applications and networks have best-effort services. They adapt the quality to the resources that are currently available.

Several categories for resource management tasks and QoS processing steps have been identified in the literature [HUT94, VOG95] such as QoS specification and mapping, QoS negotiation, resource reservation and admission control, scheduling, policing, monitoring and finally adaptation. Due to the fact that guaranteed services are less commonly available end-to-end QoS adaptation is important. QoS monitoring is essential for adaptive applications. In the next section we describe certain QoS control mechanisms. In section 5.3 we describe the dynamic QoS control mechanism in MPS using information in RTCP messages.

5.2 QoS Control Mechanisms

Research on QoS in multimedia applications has been important due to the fact that existing notion of QoS did not suffice current requirements of advanced multimedia applications. Vogel et al. [VOG95] and Hutchison et al. [HUT94] give overviews of QoS in distributed multimedia systems. QoS has been discussed in the literature at various levels. Apteker et al. [APT95] and Steinmetz and Engler [STE93] discuss QoS at the user level. Staehli et al. [STA93] address QoS at the presentation or application level. Zhang et. al [ZHA93] describe the RSVP protocol to address QoS at the resource level.

In multimedia applications best effort approaches are commonly used to control QoS. In the continuous media player [ROW92] and in Vosaic [CHE96] best effort approaches are used to adapt the quality of their presentations to the amount of resources that are

currently available. Similar approaches are discussed in [CEN95]. Approaches that provide statistical guarantees by specifying a range of acceptable quality parameters have been discussed by Ferrari et al. [FER90]. Resource management tasks such as admission, negotiation, renegotiation, scheduling, and buffer management have been discussed in [CAM93, KES95]. QoS architectures to provide end-to-end QoS guarantees by integrating these tasks have been addressed in [BAN96, NAH96].

QoS adaptation requires some kind of feedback mechanism. Many best effort applications use feedback mechanisms to adapt the rate at which the source sends, for instance video frames to the rate at which they are eventually displayed [ROW92, CEN95, CHE96]. The rate is controlled by adaptation based on the information on currently available resources. In [CEN95] a feedback toolkit approach has been described based on control system theory. A scalable feedback control mechanism for video sources has been proposed in [BOL94]. This mechanism is end-to-end and defines network state according to feedback information from receivers. This approach uses a probabilistic polling mechanism and a randomly delayed reply scheme. This requires explicit probing. Feedback controller presented in [KAN93] requires the switches to send buffer occupancy and service rate information to the source. In [BUS96] a mechanism for dynamic QoS control based on RTP is described. This is similar to the approach used in our system as discussed in the next section.

5.3 Dynamic QoS Control in MPS

Our approach to QoS control is end-to-end and is similar to that described in [BUS96]. This approach uses feedback provided through RTCP RR (figure 3.14) messages to dynamically control QoS parameters such as bandwidth usage and frame rate at the sender. Based on the information provided in RR message, current network state is estimated, current bandwidth availability is calculated and the transmission rate at the source is controlled accordingly.

$P_e[i] = S_h - S_f$ $P_l[i] = P_e[i] - P_r[i]$ $P_{e(i, i-1)} = P_e[i] - P_e[i-1]$ $P_{r(i, i-1)} = P_r[i] - P_r[i-1]$ $P_{l(i, i-1)} = P_{e(i, i-1)} - P_{r(i, i-1)}$ $P_{lf(i, i-1)} = P_{l(i, i-1)} / P_{e(i, i-1)}$ $P_e[i-1] = P_e[i]$ $P_r[i-1] = P_r[i]$ $P_l[i-1] = P_l[i]$	$P_e[i]$ - Number of packets expected at ith RR S_h - Highest Sequence number received S_f - First Sequence number received $P_r[i]$ - Number of packets received at ith RR $P_l[i]$ - Number of packets lost at ith RR $P_{lf(i, i-1)}$ - Fraction of packets lost during two consecutive RRs
--	---

Figure 5.2: Packet Loss Estimation at the Receiver

QoS control is crucial for the case of video as the transmission requirements for video are very high and can easily overload the network. In a video pipeline from source to the destination there are several resources that are required such as network bandwidth, buffers at the source, switches and destination. In a congested pipeline, packets may be delayed or lost, or the destination may not be able to decode a frame due to CPU

congestion. Thus the resources utilized in earlier stages are wasted on a frame that is eventually dropped at the receiver. Hence network state estimation is an important aspect in QoS control.

RR message contains packet loss and packet delay jitter information. Packet losses occur either due to bit errors or due to buffer overflow at the switches or network interfaces. Network congestion is the most common reason for packet losses. Packet loss indicates current congested network state while packet delay jitter indicates a forthcoming congestion state. Delay jitter estimation is described in chapter 4 (figures 4.5 and 4.6). We use packet loss as definite network state indicator. RX agents estimate packet loss as

$L = (1-C)*L + C*F_l$	L - smoothed loss
$\text{if } L > L_h, \text{ Network State} = OL$	C - lowpass filter constant
$\text{if } L < L_l, \text{ Network State} = UL$	L_h, L_l - high & low loss thresholds
$\text{if } OL, B_a = \max \{ B_c * u, B_{min} \}$	OL, UL - over loaded, under loaded network state
$\text{if } UL, B_a = \min \{ B_c + v, B_{max} \}$	u - multiplicative decrease factor
	v - additive increase factor
	B_{min}, B_{max} - min. & max. bandwidth
	B_c - current bandwidth used
	B_a - adjusted bandwidth available

Figure 5.3: QoS Control Steps in MPS

described in [SCH96a]. Figure 5.2 shows packet loss estimation by the RX agents. P_l indicates the cumulative packet loss and P_{lf} indicates the fraction of packets lost since the last RR message was sent.

The TX agents follow the steps summarized in figure 5.3 to control QoS. Upon receiving RR messages from RX agents, statistics of packet loss are estimated. The current fraction of packet loss value indicated in the RR message is passed through a lowpass filter to minimize QoS oscillations in the system. Selection of filter constant, C is important since increasing C increases the influence of the input to the filter which is the current value of fraction of packets lost and decreasing C results in a higher influence on the old values.

The output of the lowpass filter is smoothed packet loss L . This value is compared to the low and high loss threshold values, L_l and L_h to estimate the network state as over-loaded or under-loaded, OL , UL . The minimum and maximum bandwidth available, B_{min} and B_{max} , for the TX agents can be configured at start up based on the underlying network used. The available bandwidth that is adjusted to B_a is calculated following a well-known mechanism which is also used in ATM rate-control of ABR traffic [SAT94] and feedback control of video distribution [BOL94]. Two factors u , v are used to increase or decrease the current bandwidth used, B_c to estimate B_a . When the network state is over-loaded, we use multiplicative decrement factor u to rapidly decrease the bandwidth available and when the network state is under-loaded we use additive increment factor v to gradually increase the bandwidth available.

The efficiency of this QoS control mechanism largely depends on the choice of several parameters such as filter constant, loss threshold values and increment, decrement factors. Experiments conducted by Busse et al. [BUS96] indicate that a value of 0.3 for filter constant works well both in the internet and local ATM network, decrement factor of

0.875, increment factor of 50 and threshold values of 5% and 10% are suggested. Threshold values should be set such that the scheme is aggressive enough to exploit the available bandwidth. The current rate at which the video TX agent sends out frames is adjusted based on two factors, user specified frame rate and the available bandwidth, favoring the most restrictive metric.

Chapter 6

Implementation

6.1 Hardware and Software Environment

The hardware environment for the development of our real-time Multimedia Presentation System (MPS) consists of a set of SUN Sparc-20 workstations that served as a server and a client (figure 6.1). The machines are connected by a 10Mbps Ethernet.

Figure 6.1: Layout of Workstations in MPS

The operating system for the workstations is UNIX running under Solaris version 2.5. SunVideo card along with SunVideo tool are used to digitize audio and video pre-recorded from television news segments. The programming languages used to develop server software are GNU's C, C++ version 2.7.2 and Tcl version 7.5. The motion-JPEG video was produced using SunVideo card on the server machine. The client software is developed using GNU's C, C++ version 2.7.2, Tcl 7.5/ Tk 4.1 [OUS94]. Some modules from the implementation of vic [MCA95] have been adapted for use in MPS. UNIX socket system calls [STEV90] are used for communication between client-server modules. The code is compiled using GNU's gcc version 2.7.2.

6.2 Software Architecture

The software architecture of our system is based on object-oriented framework with a goal to achieve flexibility. This allows to easily modify and extend the code so as to explore new coding schemes, network models and compression methods of various media types. At present the network model supported by MPS for media transport is UDP/IP. The compression methods supported for video is motion-JPEG. Additional modules can be built and integrated easily when object-oriented framework is used.

Object-oriented software development is fundamentally different from traditional software development because the object-oriented paradigm allows real-world objects to be modeled by corresponding programming objects. Traditional programming environments

are characterized by procedural programming paradigms which tend to model real-world tasks by procedures. Object-oriented paradigm is a data flow paradigm while traditional techniques are control flow.

The other advantages in using object-oriented framework result from its PIE principle. PIE principle that stands for polymorphism, inheritance and encapsulation is the basic concept of object-oriented environments. Polymorphism allows the same operation to be performed by different objects. The behaviour depends on the operation of an object that can be bound dynamically. With inheritance the base classes can be extended to form sub classes with additional data and operations defined on them. Encapsulation groups data and operations within type definitions. The benefit of the PIE principle is that it maximizes software re-use by incremental development thereby allowing software components to grow from existing components and reducing the time to create these new components.

Basing the software architecture on object-oriented framework allows our system to take advantage of the above described features. The software architecture is built upon an event-driven model that makes use of the Tcl/Tk framework. A set of base classes are implemented in C++ and are coordinated through Tcl/Tk. The approach of implementing low-overhead control functionality in Tcl and performance critical data handling in C++ is used. It allows for rapid prototyping without degrading system performance.

6.2.1 MPS Server

The MPS server is concurrent and client-driven, that is all data transmitted from the server to the client is explicitly requested. Two types of server processes are used to achieve concurrency. The master server process (MSP) accepts incoming connections at a well-known port and creates a slave server process (SSP) using *fork* system call to service the client. The slave handles all the client requests and closes the connection when the client is finished. The algorithm for concurrent MPS server is shown in figure 6.2.

Algorithm for concurrent MPS server

Master Server Process (MSP)

1. Create a socket and bind to well-known address. Leave the socket unconnected.
2. Call `select` to check for client request.
3. If a request is received, call `fork` to create a slave process.

Slave Server Process (SSP)

1. Receive specific request and communicate with client using new socket.
2. Create a socket for media IPC (inter process comm.)
3. Create a pair of sockets for video transport and feedback.
4. Create an audio server process (ASP), call `fork` and `exec`.
5. Service all requests of the client and exit.

Audio Server Process (ASP)

1. Create a socket for media IPC (inter process comm.)
2. Create a pair of sockets for audio transport and feedback.
3. Service all requests and exit with SSP.

Figure 6.2: Algorithm for Concurrent MPS Server Creation

The slave process embraces the functionality of both SCM and video TX agent components that are shown in figure 3.2. The functionality of audio TX agent (figure 3.2) is handled by a separate audio server process (ASP). This is created by the slave process

by forking itself and calling *exec* system call. The advantages of having audio and video handled by separate processes are already discussed in chapter 3. The process oriented structure along with the external interfaces such as the network is shown in figure 6.3. The slave process and the audio process created by it form a set of collaborating processes that communicate via sockets to carry out the client requests.

Figure 6.3: Process Oriented Structure of MPS Server

Figure 6.4 shows the class hierarchy in SSP server process. The classes on the left form the base classes. The derived classes are shown to the right of the base classes. A small square is used to denote multiple inheritance point. For instance, Transmitter class is derived from Timer and TclObject classes. The standard technique of having classes inherit from a TclObject class is used. This allows all its subclasses to have an interface between Tcl and C++ by a command method. The class hierarchy in ASP is similar to that in SSP (figure 6.5).

Figure 6.4: Class Hierarchy in Slave Server Process (SSP)

The AppManager class captures the interface for Session Control Protocol (SCP) implementation. The classes that contain methods to read data from video file are Mjpg_Ctrl_File, Mjpg_Data_File and MediaFileGrabber. To read data from PCM audio file Media_Data_File and MediaFileGrabber are used. The

Figure 6.5: Class Hierarchy in Audio Server Process (ASP)

PCM_Encoder class (figure 6.5) and JpegFramer class encode the audio and video data into RTP packets and hand then to respective MediaSessionManager. The VideoSessionManager class together with RtpCtrlHandler class have methods to carry out the RTP functions for video; similarly AudioSessionManager class is defined to carry out RTP audio session. Ipc_Channel and MediaBus classes have methods to establish inter-process communication. IPNetwork class encapsulates the necessary functions to send and receive data using sockets.

6.2.2 MPS Client

In many aspects the MPS client shares similar ideas that are used in creating MPS server. The algorithm for MPS client creation is summarized in figure 6.6. The master client

Figure 6.6: Algorithm for MPS Client Creation

process (MCP) initiates a separate audio client process (ACP). The MCP calls `fork` system call upon startup followed by `exec` system call to initiate ACP. The MCP embraces the functionality of both SCM and video RX agent components shown in figure 3.2. The functionality of audio RX agent is handled by ACP. The process oriented structure consists of simply two processes communicating via sockets (figure 6.7).

Figure 6.7: Process Oriented Structure of MPS Client

The class hierarchy of MCP is shown in figure 6.8. The base classes appear on the left and the derived classes are towards their right. A small square is used to denote the multiple inheritance. The standard technique of having classes inherit from `TclObject` class is used just as in the case of server processes. This allows all its subclasses to have an interface between `Tcl/Tk` and `C++` by a `command` method. The class hierarchy in ACP is shown in figure 6.9. The base classes in both cases are mostly common.

The `AppManager` class at the client has interface that captures user interactions, translates them into commands and communicates them to the server using `Session Control Protocol`. Other than `AppManager` the classes with the same name at server and client carry similar functions. For instance, `IPC_Channel` and `MediaBus` classes at

Figure 6.8: Class Hierarchy in Master Client Process (MCP)

both server and client have methods to establish inter-process communication.

IPNetwork class has the interface to carry out send and receive functions using sockets.

Figure 6.9: Class Hierarchy Structure in Audio Client Process (ACP)

The reassembly and decoding of motion-JPEG video is achieved through `JpegReassembler`, `MJpegDecoder` while rendering is done using `VideoWindow` and `WindowRenderer` classes. Similarly, for PCM audio the classes that reassemble and decode audio data are `AuRessembler`, `PCMDecoder`. Audio is rendered using `Renderer` and `SUNAudio` classes. In the next section details pertaining to the user interface at the client and the operations are discussed.

6.3 User Interface and Mode of Operation

The first prototype of our system's graphical user interface (GUI) , developed using `Tcl7.5/Tk4.1` is shown in figure 6.10. `Tcl` stands for “tool command language”; it is a simple scripting language for controlling and extending applications. It is also embeddable since its interpreter is implemented as a library of C procedures that can easily be incorporated into applications. The core `Tcl` features can be extended with additional commands specific to the application.

`Tk` is one of the most useful extensions to `Tcl`. It is a toolkit for the X Window System with extensions to the core `Tcl` commands for building user interfaces. `Tk` is also implemented as a library of C procedures so it can be used in applications. Base `Tk` features can be extended with new user-interface widgets. The main benefits of using `Tcl` and `Tk` are rapid development, user convenience and glueable language.

The menus appear at the top of the GUI and various panels appear below the menu bar (figure 6.10). The user can play only audio, only video or both media. One of the objectives in our system design is to support user interactivity during playback. Our user-friendly interface has media control, volume control and rate control panels. It also has a session status panel to indicate the length of the clip and to show the percentage of clip that is received and played.

Figure 6.10: User Interface of MPS

The different widgets used to construct the various components of the GUI are *frame*, *toplevel*, *label*, *separator*, *button*, *checkboxbutton*, *radiobutton*, *menu*, *menubutton*, *entry* and *scale*. *Frame* is used to hold other widgets in different panels and the menu bar. *Label*

widget is used wherever text is to appear. A new widget to depict progress bar and volume indicator is created using a combination of *frames*. The *entry* widget is used to facilitate the user to enter the title of the clip. The *menu*, *menubutton*, *checkbutton*, *radiobutton*, and *separator* are used to create the components of the menu bar. The *button* widget is used to create the various command buttons in media control and volume control panel including play, pause, rewind, forward, stop, mute/unmute and volume increase/decrease. The *scale* widget is used in rate control panel to facilitate the user to indicate the desired frame rate. The *toplevel* widget is used to create the video window that shows video in a separate window.

Once the user has selected a clip and started playing, the user can pause, forward or rewind and resume playback or stop playing the present clip. Any number of clips can be played sequentially in a single MPS session. Thus the client need to connect only once to the server no matter how many clips it requires to play.

Figure 6.11 shows a snapshot of the screen when the user has stopped playing a clip. The video window shows a *cnn* clip that was played. We could achieve frame rates up to 25 fps under light traffic and workstation load conditions when played on a Sparc-20 client workstation connected to a different Sparc-20 server workstation over an Ethernet. The session status panel indicates the length of the clip as 50 seconds and the progress bar indicates the approximate position of the length of the clip in percentage that is retrieved and played at the client workstation.

Figure 6.11: A Snapshot of MPS Showing Cnn Video Clip

In figure 6.12 a scenario of simple MPS session with one MPS client and one MPS server is shown. From left to right the GUI, main client process (MCP), master process at the server (MSP), slave process at the server (SSP), audio process at the server (ASP) and audio process at the client (ACP) are shown. The scenario in figure 6.12 indicates the interaction sequences in a session where the user selects only audio media type, plays, stops and exits.

Figure 6.12: Interaction Sequence Diagram for Audio Only Scenario

6.4 Experimental Results

In this section we present the results of the experiments we carried on LAN and over the Internet environments. The test data consisted of clips recorded from mainly television news segments; the details of the test data are tabulated in table 6.1. The frame rates at which the clips were digitized ranged from 16 to 20 fps. The pixel resolution of all the video clips is 320 by 240. The sampling rate of all audio clips is 8000 samples per second where each sample consists of 8 bits or 1 byte. The length of the clips ranged from 33 to 60 seconds. The user need to specify the clip title when requesting playback of any media pertaining to the specified clip title.

Table 6.1: Video and Audio Test Data Details

Clip Title	rail	cnn	cbc
File Name	rail.Jpeg	cnn.Jpeg	cbc.Jpeg
Record Rate (fps)	18	16	20
Image Size	320 X 240	320 X 240	320 X 240
No. of Frames	620	1000	1000
Length (sec)	33	61	50
File Name	rail.au	cnn.au	cbc.au
Sampling Rate	8000	8000	8000
Data Size (bytes)	270336	492544	405504
Length (sec)	33	61	50

In table 6.2 the names of the hosts used for our experiments and their functions are summarized. The client side of all our tests is based in our laboratory in Carleton University. In order to conduct experiments on various network configurations, we set up servers on LAN as shown in figure 6.1 and at the MIRL (Multimedia Information Research Laboratory) in the University of Ottawa for the case of the Internet. The remote server is a Sparc-20 workstation running Solaris 2.4.

Table 6.2: Hosts and their Functions in our Experiments

Node	Function
vanguard.sce.carleton.ca	Local Client
pioneer.sce.carleton.ca	Local Server
rigel.genie.uottawa.ca	Internet Server

In figure 6.13 a graph shows the combined video decode and render times that were collected on a lightly loaded client workstation. The decoding and rendering of motion-JPEG video is done in software; the modules have been adapted from vic[MCA95] which carry their origins from the efforts of the independent JPEG group. The clip used is *iscbc*.

Figure 6.13: Decode and Render Times of Cbc M-JPEG Video Frames

The decode and render times largely depend on the content of the video and the motion that may be involved. In the graph shown in figure 6.13 at frame number 50, 420 and 750 an increase in the time taken to decode and render can be noted, this indicates a change in the scene at that frame. In between the frame numbers there is relatively less motion and hence the decode and render times vary slightly as can be inferred from the graph. After frame 750 the decode and render times are higher than 50 milliseconds indicating that there is significant motion involved in the video. The jitter in video decode and render is within the level of human perception of synchronization. It is observed from the graph that overall decode and render times vary over a range of about 100 milliseconds whereas between frames the variation in time is much smaller.

6.4.1 Experiments over LAN and Analysis

Experiments on LAN were conducted between *vanguard* and *pioneer* in our laboratory at Carleton University. In table 6.3 we show the results of our tests on video conducted on LAN. Audio tests did not show degradation in quality when tested over LAN. So we did not tabulate audio results. The bandwidth requirements for audio is only 64 Kbps which is significantly low when compared to that required for video. In our experiments on LAN the client and server were on the same subnet. The numbers in table 6.3 give the total number of packets, the frame rate at which they were sent which can be set by the user at the client, the bandwidth setting at the server, the total number of packets received, the total number of frames received, the total number of packets missing, the percentage of packets lost and the percentage of frames lost.

Table 6.3: Video Test Results on LAN

Clip Title	rail	cnn	cbc	
File Name	rail.Jpeg	cnn.Jpeg	cbc.Jpeg	cbc.Jpeg
Frame Rate (fps)	12	16	20	14
Max. Bandwidth (Kbps)	865	955	1500	1010
Total No. of Packets sent	5404	11823	7037	7037
No. of Packets Received	5350	11821	7037	6955
No. of Packets Missing	56	2	0	83
No. of Frames Received	617	999	1000	993
No. of Frames Missing	3	1	0	7
% loss (packets)	1.03	0.01	0.0	1.17
% loss (frames)	0.4	0.1	0.0	0.7

There are several inferences that can be made from the above results. Although the total number of frames in *cnn* and *cbc* clips is 1000, the total number of packets in *cnn* is much higher. The total number of packets in each frame depends on the image content of the video clips. Recall from chapter 3 that each video frame is segmented into multiple number of RTP packets; the length of each RTP packet is 1024 bytes. The frame rates shown in the table were achievable with the given maximum bandwidth settings; the frame rate received was the frame rate requested. In the middle two columns showing the results for *cnn* and *cbc* it can be noted that there is insignificant number of missing packets; there was little or no other network traffic and nearly all packets are received by the client.

The results in the first column corresponding to *rail* video clip and the last column corresponding to *cbc* video clip were collected by inducing equal traffic load between *vanguard* and *pioneer*. It can be noted that in these two cases the total number of packets lost is noticeable, however the total number of frames received is still a good number. The percentage of both packet and frame losses is not more than 2 percent. Overall the quality of video appeared to be satisfactory to the end user.

Figure 6.14: Relative Transit Delay for Cbc Video Frames on LAN

In figure 6.14 a graph with relative transit delays in *cbc* video clip comprising 1000 video frames is shown. The delays are in the range of ± 50 ms with majority in the range of ± 10 ms. Relative transit delay indicates how far ahead or how far beyond a video frame is received than its expected arrival time. Figure 6.15 shows a companion graph

indicating the application level inter-frame-arrival jitter. The jitter is estimated as described in chapter 4. These two graphs are for the *cbc* video clip when both audio and video were selected for playback at the client.

It can be observed that after frame number 750 there is a rise in relative transit delay as well as in inter-frame-arrival jitter. This can be explained by looking at the graph in figure 6.13 showing the decoding and rendering delay for the same video clip; the fact is the

Figure 6.15: Inter-Frame-Arrival Jitter for Cbc Video on LAN

frames after 750 consist of more image data and hence the frame is longer in terms of number of packets than the previous frames in the same clip. A longer frame takes more time to arrive at the client than a shorter frame and hence the jitter increases momentarily. On an average the jitter is still below 20 ms which is within the limits of human perception of synchronization of audio and video.

In figure 6.16 the relative transit delay in audio packets of *cbc* audio clip is shown. The audio packets are of 40 millisecond duration. The total number of audio packets in this

Figure 6.16: Relative Transit Delay for Cbc Audio Packets on LAN

clip are 1268. The server sends audio packets at a constant rate of 25 packets per second to make up for constant audio rate of 8000 bytes per second or 64 Kbps. The relative transit delay is in the range of ± 10 milliseconds but the majority is only within ± 1 millisecond. In figure 6.17 the inter-arrival jitter experienced by the same packets is shown. The estimation of jitter is detailed in chapter 4. From the graph in figure 6.17 it is evident that the jitter experienced by audio packets is almost a constant within 1 millisecond.

Figure 6.17: Inter-Arrival Jitter for Cbc Audio Packets on LAN

The analysis of the above experiments on LAN showed that the audio packets experienced jitter which is not very significant and video experienced jitter which is still within the limits to achieve synchronization with audio. The occasional packet losses in video clips did not affect the video playback quality to a significant extent.

6.4.2 Experiments over Internet and Analysis

In this section we present the results of our experiments conducted in the Internet environment. We used *vanguard* as client and *rigel* located in University of Ottawa as the server. The distance between the two machines is measured using a *traceroute* utility. It is shown that *rigel* and *vanguard* are away by a maximum of 30 hops. The results of *traceroute* from *vanguard* to *rigel* are shown in figure 6.18.

```

traceroute to rigel.genie.uottawa.ca (137.122.20.177),
30 hops max, 40 byte packets
 1 galaxia (134.117.4.3) 2.423 ms 1.111 ms 1.477 ms
 2 gate-galaxia (134.117.61.1) 2.386 ms 2.253 ms 2.472 ms
 3 mell-gate.carleton.ca (134.117.11.1) 23.808 ms 3.023 ms 3.312 ms
 4 carleton.onet.on.ca (134.117.18.2) 4.513 ms 3.929 ms 4.233 ms
 5 ottawa3-carleton-if.onet.on.ca (130.185.17.9) 12.864 ms 21.162 ms 23.634 ms
 6 uottawa-ottawa3-if.onet.on.ca (130.185.17.14) 21.686 ms 37.281 ms 34.006 ms
 7 here.wall.uottawa.ca (192.75.139.1) 118.445 ms 55.648 ms 67.269 ms
 8 drogw.uottawa.ca (137.122.62.2) 67.474 ms 79.875 ms 18.937 ms
 9 rigel.genie.uottawa.ca (137.122.20.177) 80.205 ms 52.144 ms 17.743 ms

```

Figure 6.18: Results of Traceroute from Vanguard to Rigel

In table 6.4 the results of video tests over Internet are summarized. Due to the dynamic nature of the traffic on the Internet it was not easy to get accurate estimates of the statistics. The results shown in table 6.4 were collected during moderate and busy usage hours. It can be noticed that video data suffered high percentage of both packet and frame losses. Low frame rates were observed even with high bandwidth settings at the server.

Table 6.5 shows audio test results over the Internet when the user selects audio only and both audio and video for playback. The audio playback quality was noted to be satisfactory to the end user when only audio media type was selected. The occasional packet losses did not degrade the playback quality. However when the user selects both audio and video for playback the loss statistics increase due to insufficient bandwidth availability. The audio packet losses for *rail* clip when both media are requested is the lowest; this is due to 8 fps of video frame rate that was requested. In the worst case audio quality was poor but was still tolerable.

Table 6.4: Video Test Results over Internet

Clip Title	rail	cnn	cbc
File Name	rail.Jpeg	cnn.Jpeg	cbc.Jpeg
Frame Rate (fps)	4	8	2
Max. Bandwidth (Kbps)	725	865	150
Total No. of Packets sent	5404	11823	7037
No. of Packets Recvd.	4594	9459	5111
No. of Packets Missing	810	2364	1927
No. of Frames Recvd.	514	618	527
No. of Frames Missing	106	382	473
% loss (packets)	15	20	27
% loss (frames)	17	38	47

Table 6.5: Audio Test Results over Internet

Clip Title	rail	cnn	cbc
File Name	rail.au	cnn.au	cbc.au
Total No. of Packets sent	845	1540	1267
No. of Packets Recvd.	799	1483	1197
No. of Packets Missing	46	57	70
% loss (packets)	5.4	3.7	5.5
File Name	rail.au	cnn.au	cbc.au
Total No. of Packets sent	845	1540	1267
No. of Packets Recvd.	686	1240	1137
No. of Packets Missing	159	300	130
% loss (packets)	18.8	19.4	10.2

Figure 6.19: Relative Transit Delay for Cbc Video Frames over Internet

In figure 6.19 a graph showing the relative transit delay for *cbc* video frames as observed by the client is shown. The relative transit delay in this case varies widely in the range of ± 75 ms with majority in the range of ± 25 ms. It indicates that the frames need to be

Figure 6.20: Inter-Frame-Arrival Jitter for Cbc Video over Internet

adequately buffered at the client for playback in order to remove the inter-frame-arrival jitter (figure 6.20). The average jitter is between 30-35 ms which is higher than that observed over LAN.

Figure 6.21: Relative Transit Delay for Cbc Audio Packets over Internet

Figures 6.21 and 6.22 show the relative-transit delay and inter-arrival jitter for audio packets of *cbc*. The relative transit delay is concentrated in the range of ± 20 ms. The inter-arrival jitter in audio is observed to be around 10 ms on an average.

The results of the experiments on Internet reveal that due to unpredictable load on the network high video frame rates could not be achieved; however low frame rate video can be achieved. Audio over Internet is possible due to low bandwidth requirements of audio media type. When both audio and video are desired there is significant bandwidth requirement demands placed on the Internet. Again due to the unpredictable nature of

Figure 6.22: Inter-Arrival Jitter for Cbc Audio Packets over Internet

specific usage of the Internet this may be achieved with variable jitter in both inter-frame arrival for video and inter-packet-arrival of audio.

Chapter 7

Conclusions

7.1 Summary

In this thesis we presented the design, implementation of a real-time Multimedia Presentation System (MPS) and the experimental results achieved. The aim of this work was to investigate media-on-demand issues using Real-time Transport Protocol (RTP) over different network infrastructures. In order to realize this objective we described our system architecture, illustrated the communication protocols, suggested the synchronization mechanism and adaptive QoS control method to be applied in MPS using RTP and RTCP. We conducted several experiments over LAN and over the Internet to test the feasibility of transmitting real-time audio and video.

The system design of MPS was driven by distributed client-server and object-oriented design principles. The main features of MPS can be summarized as real-time media transmission, flexibility, scalability and user interactivity during media presentation. Real-time audio and video transmission is achieved through the use of RTP. We use RTCP to

provide feedback information on media reception quality. Separate channels are used for actual media data and control data for each media. Audio and Video are handled in a non-interleaved fashion at all the levels, source, transport and destination. This gives the flexibility to control each media type with different priorities. This also gives the flexibility to the system to support new media types. The distribution of media on multiple servers can be realized easily once they are segregated and handled separately thus allowing the system to be scalable.

We designed a Session Control Protocol (SCP) over TCP/IP for controlling the MPS session between the client and sever entities. The main functions of SCP is to support the exchange of media specific information, media delivery and control requests. This translates the user interactions at the client to the server.

A prototype of our system has been implemented and evaluated. The client and server modules of our system have been implemented using GNU's C++, C version 2.7.2 and Tcl7.5. Motion-JPEG video and PCM audio media types are currently supported. The AV player application is developed to support the user interface at the client using Tk4.1. User interactivity is achieved through media control, volume control and rate control panels. The user can select the media type as audio only, video only or both audio and video.

Several experiments were conducted both on LAN and over the Internet. Performance of our system is evaluated in the face of varying network load conditions. The test results indicate that audio can be played both over LAN and the Internet without perceivable

quality degradation. We could achieve high video frame rates on LAN when both client and server were on the same subnet. Over the Internet we could achieve frame rates of up to 8 fps of 320X240 motion-JPEG video. The inter-arrival jitter of video frames and audio packets vary over LAN and Internet depending on the traffic load.

7.2 Future Work

While the system architecture is in place there are various possible refinements and extensions in future. We support video frame rate as one of the user QoS specifications. In future we plan to incorporate more user QoS specification parameters such as various resolutions and synchronization accuracy. This work has explored the idea of media synchronization using RTP; we described the synchronization mechanism using the RTP timestamp information while no prior work on RTP has addressed the issue of synchronization in a greater detail. Future research is required to further refine and evaluate these algorithms under varying network as well as workstation load conditions.

Under varying workstation load conditions, the real-time requirements are not met by the traditional time-sharing operating-systems. The lack of real-time support can have a profound effect on media processing at the workstations which can affect the delay estimation, crucial for implementing synchronization algorithms. These issues need further investigation and research.

Our system currently supports motion-JPEG video and PCM audio media types. While the compression achieved with motion-JPEG is not as high as MPEG it has other advantages. Since motion-JPEG is intra-frame encoded, loss of a frame does not affect the decoding of subsequent frames, whereas MPEG is inter-frame encoded and loss of I frame has a profound effect on decoding P and B frames. Although using motion-JPEG has the above advantages the most glaring disadvantage is its bandwidth requirement. For this reason, support for MPEG media type is a possible future extension.

Only software decompression of motion-JPEG is currently supported. One of the disadvantages of software decompression is the reliance on computational resources of the local host. When decoding cards are available on the machine, support for hardware decompression has to be added to the current system. Software decompression takes a lot of CPU processing time and under heavy load conditions this saturates the CPU resulting in quality degradation. To address this problem, we need algorithms to adapt available CPU resources by shedding load and gracefully adapting to load fluctuations.

At present we support UDP/IP for unicast transmission of real-time media using RTP. Since RTP supports multicast we need to enhance our system to support multicast for applications like tele-learning. RTP is end-to-end and is independent of underlying network infrastructure. By adding a class that encapsulates ATM network model, we can extend the system to transmit media over both UDP/IP and ATM. This is useful when high quality audio and video with guaranteed end-to-end delays are desired.

The other area that requires more attention is the protocols. For carrying motion-JPEG using RTP, we fragment each frame, append the RTP header and JPEG header to each packet of a frame. There is some redundancy that is incurred in this process for pre-recorded media. This needs further refinement and drafts to specify a standard means of carrying motion-JPEG video frames over any type of network. We use RTP over UDP/IP instead of more reliable TCP/IP for media transport as a natural way to carry real-time data following the argument that retransmission and congestion control methods in TCP are not well suited for real-time data as it increases the delay. Following the same reasoning, it is arguable whether to use TCP or UDP for session control protocols like our SCP. Recently, this has been one of the topics in the discussion concerning the RTSP protocol which is still evolving.

We addressed the media-on-demand issues over the Internet with a focus on real-time media. It is also desired to support other media types like text and graphics along with audio and video. It is a future work item to support all media types using a common interface like the world wide web (WWW) client interface. In conjunction we would also like to support a WWW server along with a Java-based client.

Appendix A

Publications

1. S. Palacharla, A. Karmouch and S.A. Mahmoud, "Design and Implementation of a Real-time Multimedia Presentation System using RTP", *Proc. of IEEE COMPSAC ' 97* (to appear), Washington, DC, Aug. 1997.

Bibliography

- [ALL83] James F. Allen, "Maintaining Knowledge about Temporal Intervals", *Communications of ACM*, Vol. 26, No. 11, 1983.
- [APT95] R. T. Apteker, J. A. Fisher, V. S. Kisimov and H. Neishlos, "Video Acceptability and Frame Rate", *IEEE Multimedia*, Vol. 2, No. 3, pp. 32-40, Fall 1995.
- [BAN96] A. Banerjea, et al., "The Tenet Real-Time Protocol Suite: Design, Implementation and Experiences", *ACM Transaction on Networking*, Vol. 4, No. 1, pp. 1-10, Feb. 1996.
- [BLA96] G. Blakowski and R. Steinmetz, "A Media Synchronization Survey: Reference Model, Specification, and Case Studies", *IEEE JSAC*, Vol.14, No. 1, pp. 5-35, Jan. 1996.
- [BOL94] J. C. Bolot, T. Turetti, and I. Wakeman, "Scalable feedback control for multicast video distribution in the internet", *ACM SIGCOMM*, pp. 58-67, London, England, Aug. 1994.
- [BUL91] D.C.A. Bulterman and R. Van Liere, "Multimedia Synchronization and Unix", *Proc. of Int'l Workshop on Network and Operating System Support for Digital audio and Video*, pp. 108-119, Heidelberg, Germany, Nov. 1991.
- [BUS96] I. Busse, B. Deffner and H.Schulzrinne, "Dynamic QoS Control of Multimedia Applications based on RTP", *Computer Communications*, Jan. 1996.
- [CAM93] A. Campbell, et al., "A Multimedia Enhanced Transport Service in a Quality of Service Architecture", *Workshop on Network and Operating System Support for Digital Audio and Video*, Lancaster, England, Nov. 1993.
- [CCI88] CCITT Recommendation I.350: General Aspects of Quality of Service and Network Performance in Digital Networks, including ISDN, Blue Book, 1988.
- [CEN95] S. Cen, et al., "A Distributed Real-Time MPEG Video Audio Player", *Proc. of Int'l workshop on Network and Operating Systems Support for Digital Audio and Video*, Durham, NH, Apr. 1995.
- [CHE96] Z. Chen, S. Tan, R. Campbell and Y. Li, "Real-Time Video and Audio in the World Wide Web", *World Wide Web Journal*, Vol. 1, Jan. 1996.

- [CLA90] D. Clark and D. Tennenhouse, "Architectural considerations for a new generation of protocols", *Proc. of ACM SIGCOMM '90*, Philadelphia, PA, Sep. 1990.
- [COM93] D. E. Comer and D. L. Stevens, *Internetworking with TCP/IP Vol III: Client-Server Programming and Applications*, NJ: Prentice Hall, 1993.
- [DEE91] S. Deering, *Multicast Routing in a Datagram Internetwork*, PhD thesis, Stanford University, Dec. 1991.
- [EHL94] L. Ehley, B. Furht and M. Ilyas, "Evaluation of Multimedia Synchronization Techniques", *Proc. of IEEE Int'l Conf. on Multimedia Computing and Systems*, pp. 514-519, Boston, MA, May 1994.
- [EME93] J. Emery and A. Karmouch, "A Multimedia Document Architecture and Rendering Synchronization Scheme", *Proc. of Broadband Islands '93*, pp. 59-67, Athens, Jun. 1993.
- [ESC92] J. Escobar, D. Deutsch and C. Patridge, "Flow Synchronization Protocol", *Proc. of IEEE Globecom '92*, pp. 1381-1387, 1992.
- [FEN96] W. Fenner, R. Frederick and S. McCanne, RTP payload format for Jpeg-compressed video, internet draft (work-in-progress) *draft-ietf-avt-jpeg-02.txt*, May 1996.
- [FER90] D. Ferrari and D. C. Verma, "A Scheme for Real-Time Channel Establishment in Wide-Area Networks", *IEEE JSAC*, Vol. 8, No. 3, pp. 368-379, Apr. 1990.
- [FER92] D. Ferrari, "Design and application of a delay jitter control scheme for packet-switching internetworks", *Proc. of Int'l Workshop on Network and Operating System Support for Digital Audio and Video*, pp. 72-83, 1992.
- [FRE] R. Frederick, *Network Video (nv)*, Xerox Palo Alto Research Center, available from <ftp://ftp.parc.xerox.com/net-research>.
- [FUR94] B. Furht, "Multimedia Systems: An Overview", *IEEE Multimedia Magazine*, pp. 47-59, Spring 1994.
- [HEP94] P. Hepp, et al., "Enlaces: A Multimedia Based Educational Network", *ACM Multimedia Systems*, pp. 329-336, San Francisco, CA, Oct. 1994.
- [HUT94] D. Hutchison, et al., "Quality of Service management in distributed systems", *Network and Distributed Systems*, Addison-Wesley, editor M. Sloman, pp. 213-302, 1994.

- [ITU96] ITU-T Recommendation H.323: Visual telephone systems and end equipment for local area networks which provide non-guaranteed quality of service, 1996.
- [JAC92] V. Jacobson and S. McCanne, *Visual Audio Tool*, available from <ftp://ftp.ee.lbl.gov/conferencing/vat/>, Jul. 1992.
- [KAN93] H. Kanakia, P. Mishra and A. Reibman, "An adaptive congestion control scheme for real-time packet video transport", *ACM SIGCOMM*, San Francisco, CA, pp. 20-31, Sep. 1993.
- [KAR93] A. Karmouch, "Multimedia Distributed Cooperative System", *Computer Communications*, Vol. 16, No. 9, pp. 568-580, Sep. 1993.
- [KES95] S. Keshav and H. Saran, *Semantics and Implementation of a Native-Mode ATM Protocol Stack*, internal technical memo, AT&T Bell Laboratories, Murray Hill, NJ, Jan. 1995.
- [KOU96] I. Kouvelas, et al., "Lip Synchronization for use over the Internet: Analysis and Implementation", *Proc. of IEEE Globecom '96*, London, UK, Nov. 1996.
- [LIT90] T. D. C. Little and A. Ghafoor, "Synchronization and Storage Models for Multimedia Objects", *IEEE JSAC*, Vol. 8, No. 3, pp. 413-427, Apr. 1990.
- [LI92] L. Li, A. Karmouch and N. D. Georganas, "Real-time Synchronization Control in Multimedia Distributed Systems", *Proc. of IEEE Multimedia '92*, Monterey, USA, Apr. 1992.
- [LIT91] T. D. C. Little and A. Ghafoor, "Multimedia Synchronization Protocols for Broadband Integrated Services", *IEEE JSAC*, Vol. 9, No. 9, Dec. 1991.
- [MCA95] S. McCanne and V. Jacobson, "vic: A flexible framework for packet video", *Proc. of ACM Multimedia '95*, Nov. 1995.
- [MEG94a] O. Megzari, M. B. Brahmanandam, J. Rody, G. Warnock and A. Karmouch, "MEDIABSE-A distributed Multimedia Information System", *Proc. of 17th Biennial Symp. on Comm.*, pp. 223-226, Kingston, Ontario, CA, May 1994.
- [MEG94b] O. Megzari, M. B. Brahmanandam, J. Rody, G. Warnock and A. Karmouch, "MEDIABASE- An Experiment in Multimedia Information and Communication Systems", *Broadband Islands '94*, pp. 435-438, 1994.
- [MIL92] D. L. Mills, "Network time protocol (version 3) specification, implementation and analysis", Request for comments RFC 1305, Network Working Group, Mar. 1992.

- [MIL93] G. Miller, G. Baber and M. Gilliland, "News On-Demand for Multimedia Networks", *ACM Multimedia '93*, pp. 383-392, Jun. 1993.
- [NAH96] K. Nahrstedt and J. M. Smith, "Design, Implementation and Experiences of the OMEGA End-Point Architecture", *IEEE JSAC, Special Issue on Distributed Multimedia Systems and Technology*, 1996.
- [NIC90] C. Nicolaou, "An Architecture for Real-time Multimedia Communication Systems", *IEEE JSAC*, Vol. 8, No. 3, pp. 391-400, Apr. 1990.
- [OUS94] J. K. Ousterhout, *Tcl and the Tk Toolkit*, Addison-Wesley, 1994.
- [PEN93] W. B. Pennebaker and J. L. Mitchell, *JPEG: Still Image Data Compression Standard*, Van Nostrand Reinhold, 1993.
- [RAN92] P. V. Rangan, H. M. Vin and S. Ramanathan, "Designing an On-Demand Multimedia Service", *IEEE Comm. Magazine*, pp. 56-56, Jul. 1992.
- [RAO96] A. Rao and R. Lanphier, Real Time Streaming Protocol (RTSP), internet draft (work-in-progress) *draft-ietf-mmusic-rtsp-00.txt*, Nov. 1996.
- [RAV93] K. Ravindran and V. Bansal, "Delay Compensation Protocols for Synchronization of Multimedia Data Streams", *IEEE Trans. on Knowledge and data Engineering*, Vol. 5, No. 4, pp. 574-589, Aug. 1993.
- [REI94] A. Reinhardt, "Managing the New Document", *Byte Magazine*, pp. 91-104, Aug. 1994.
- [ROD95] J. A. Rody and A. Karmouch, "A Remote Presentation Agent for Multimedia Databases", *Proc. of IEEE Int'l Conf. on Multimedia Computing and Systems*, pp. 223-230, Washington, DC, May 1995.
- [ROW92] L. A. Rowe and B. C. Smith, "A Continuous media player", *Proc. of Int'l Workshop on Network and Operating System Support for Digital Audio and Video*, La Jolla, CA, Nov. 1992.
- [SAT94] S. Sathaye, Draft ATM Forum traffic management specification version 4.0, ATM Forum 95-0013, Dec. 1994.
- [SCH92] H. Schulzrinne, *Voice Communication Across the Internet: a Network Voice Terminal*, Technical report, Dept. Of Computer Science, U. Massachusetts, Amherst, MA, Jul. 1992.

- [SCH96a] H. Schulzrinne, S. Casner, R. Fredrick and V. Jacobson, "RTP: A transport protocol for real-time applications", Request for comments RFC 1889, Internet Engineering Task Force, Jan. 1996.
- [SCH96b] H. Schulzrinne, "RTP profile for Audio and Video Conferences with Minimal Control", Request for Comments RFC 1890, Internet Engineering Task Force, Jan. 1996.
- [SHI95] N. Shivakumar, et al., "The Concord Algorithm for Synchronization of Networked Multimedia Streams", *Proc. of IEEE Int'l Conference On Multimedia Computing and Systems*, pp. 31-45, May 1995.
- [STA93] R. Staehli, et al., "Quality of Service Specification for Multimedia Presentations", *Multimedia Systems*, Vol. 3, No. 5/6, pp.251-263, Nov. 1995.
- [STE90] Ralf Steinmetz, "Synchronization Properties in Multimedia Systems", *IEEE JSAC*, Vol. 8, No. 3, pp. 401-412, Apr. 1990.
- [STE92] R. Steinmetz and T. Meyer, "Multimedia Synchronization Techniques: Experiences Based on Different System Structures", *Proc. of IEEE Int'l Workshop on Multimedia Comm.*, pp. 306-314, Monterey, CA, Apr. 1992.
- [STE93] R. Steinmetz and C. Engler, *Human Perception of Media Synchronization*, IBM European Networking Center, Technical Report 43.9310, 1993.
- [STE95] R. Steinmetz and K. Nahrstedt, *Fundamentals in Multimedia Computing and Communications*, Englewood Cliffs, NJ: Prentice Hall, 1995.
- [STEV90] W. R Stevens, *UNIX Network Programming*, Englewood Cliffs, NJ: Prentice Hall, 1990.
- [VOG95] A. Vogel, et al., "Distributed multimedia and QoS: A survey", *IEEE Multimedia*, Vol. 2, No. 2, pp. 10-19, Summer 1995.
- [WOO94] M. Woo, N. U. Qazi and A. Ghafoor, "A Synchronization Framework for Communication of Pre-orchestrated Multimedia Information", *IEEE Network*, pp. 52-61, Jan./Feb. 1994.
- [ZHA93] L. Zhang, S. Deering, D. Estrin, S. Shenker and D. Zappala, "RSVP: resource ReSerVation protocol", *IEEE network magazine*, Vol. 9, No. 5, 1993.

IMAGE EVALUATION TEST TARGET (QA-3)

APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989

© 1993, Applied Image, Inc.. All Rights Reserved