


1995

A continuous media transport protocol

Paul Michael Freeman
Iowa State University

Follow this and additional works at: <https://lib.dr.iastate.edu/rtd>

 Part of the [Computer Sciences Commons](#), and the [Electrical and Electronics Commons](#)

Recommended Citation

Freeman, Paul Michael, "A continuous media transport protocol" (1995). *Retrospective Theses and Dissertations*. 10905.
<https://lib.dr.iastate.edu/rtd/10905>

This Dissertation is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

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

A continuous media transport protocol

by

Paul Michael Freeman

**A Dissertation Submitted to the
Graduate Faculty in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY**

**Department: Electrical and Computer Engineering
Major: Computer Engineering**

Approved:

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

For the Graduate College

Members of the Committee:

Signature was redacted for privacy.

Signature was redacted for privacy.

Signature was redacted for privacy.

Signature was redacted for privacy.

**Iowa State University
Ames, Iowa**

1995

UMI Number: 9531741

UMI Microform 9531741
Copyright 1995, by UMI Company. All rights reserved.

**This microform edition is protected against unauthorized
copying under Title 17, United States Code.**

UMI

**300 North Zeeb Road
Ann Arbor, MI 48103**

TABLE OF CONTENTS

CHAPTER 1. INTRODUCTION.....	1
1.1 Benefits of Transmitting Audio and Video Data in Digital Form	3
1.2 Enabling Technologies for Interactive Television	4
1.2.1 Audio and video compression standards	5
1.2.2 Fiber optic networks.....	6
1.2.3 Digital modulation and transmission techniques	7
1.2.4 Disk storage systems	8
1.3 Protocol Requirements for Continuous Media Data Transmission	9
1.3.1 Requirement for intramedia synchronization	10
1.3.2 Requirement for ntermedia synchronization	11
1.4 Related Work	12
1.5 Organization of this Dissertation.....	13
 CHAPTER 2. VIDEO ON DEMAND PROTOCOL REQUIREMENTS.....	 15
2.1 Assumed Environment.....	15
2.1.1 Transmission of stored multimedia data only.....	17
2.1.2 Bounded transmission delays	18
2.1.3 Necessity for intermedia synchronization	20
2.2 Protocol Requirements Due to Network Architecture.....	20
2.2.1 Video On Demand Over Cable Television Distribution Networks	21
2.2.2 Video On Demand Over Telephone Switching Networks.....	30
2.3 Protocol Requirements Due to Compression Algorithms.....	32
2.3.1 Requirement for compression	32
2.3.2 The MPEG compression standard	34
2.4 Statement of Protocol Goals.....	37
 CHAPTER 3. THE CONTINUOUS MEDIA TRANSPORT PROTOCOL	 39
3.1 Protocol Operating Environment	39
3.2 CMTP Resource Optimization Strategy.....	41
3.3 Data Structures Required at the Continuous Media Server.....	43
3.3.1 The Delivery Deadline Table	44
3.3.2 The Transmission Schedule Table.....	46
3.4 Calculation of Resource Requirements.....	52
3.5 Calculation of the Transmission Schedule Table.....	58
3.6 Calculation of t_{SV}	65
3.7 Calculation of the Buffer Space Required at the Receiver	67
3.8 Calculation of B_{SV}	69
3.9 Calculation of the Minimum Transmission Rate for a Fixed Size Receive Buffer ..	70

CHAPTER 4. CMTP REFINEMENTS FOR NON-IDEAL NETWORKS	81
4.1 Protocol Refinements to Compensate for Network Jitter.....	81
4.2 Protocol Refinements to Compensate for Clock Frequency Errors.....	82
4.3 Protocol Refinements for Intermedia Synchronization.....	86
CHAPTER 5. CONCLUSION	88
BIBLIOGRAPHY.....	90

CHAPTER 1. INTRODUCTION

Imagine watching a football game from your home television set while being able to check scores from other league games, examine player statistics, or order tickets for a future game all with the push of a button. Or, while watching the nightly news, being able to selectively navigate through news stories for the day in order to obtain detailed coverage of those news events which are of particular interest, while skipping stories which aren't. Imagine having the power to personally design, and change at will, a network's television broadcast schedule so that your favorite television programs are broadcast exactly when you choose, and not when the network programmers choose.

During the last several years, the computer, telecommunications, and cable television industries have begun to focus large research efforts aimed at defining new television services such as those described above. These services are radically different from the broadcast television service which exists today. Termed "*interactive television*", this technology promises to allow subscribers the ability to personally select the movies and television shows they wish to view and specify when they wish to view them, browse on-line product catalogs and place orders for items of interest, and download and play multi-user video games against opponents located in remote cities, all from the convenience of their home television sets.

The key to interactive television is the storage, transmission, and processing of video and audio information in digital, rather than analog, form. It is the continued, steady growth in digital storage densities, network transmission bandwidths, and digital integrated circuit complexities that will combine to make digital interactive television both economically and

technologically feasible in the near future.

However, there are still challenges to be solved before widespread deployment of interactive television can become a reality. A key area where work is still required is in the design of communication protocols which support the unique requirements of digital audio and video information. Existing communication protocols are designed to provide reliable, error-free data delivery through time-out and retransmission mechanisms. The cost of such error-free data delivery is the lack of an upper bound on the end-to-end transmission delay. However, digital audio and video data impose stringent real-time constraints on a communication protocol. Once a receiver has begun play-back of an incoming audio or video data stream, the communication protocol must guarantee that the remainder of the data is received before it is scheduled to be played back. However, the data cannot be transmitted so fast that it overflows the buffer space at the receiver.

This dissertation studies the unique communication protocol requirements for digital audio and video data transmission, and proposes a transport layer protocol which can meet these requirements. The protocol proposed here is mainly designed to support a type of interactive television known as "*video on demand*" (VoD). Video on demand allows a subscriber to navigate through on-line menus, select a movie or television show of interest, and receive the selected program over a dedicated network connection. In addition to home entertainment applications, video on demand systems can be used for applications such as corporate training, hotel and hospital room video distribution, and airplane and cruise ship entertainment systems.

Although the protocol developed here is specifically designed to meet the real-time requirements for video on demand applications, it also meets many of the requirements for the communication of stored multimedia data in general.

1.1 Benefits of Transmitting Audio and Video Data in Digital Form

The storage and transmission of audio and video information in digital form offers many benefits over the more conventional approach of storing and transmitting the information in its original analog form.

- **Greater immunity to noise**

Digital signals offer greater immunity to noise than do analog signals. Even the smallest amount of noise adds distortion to an analog signal. However, digital signals can tolerate some noise as long as the noise level is not so large that it causes the value of a bit to change. Thus, the transmission of audio and video information in digital form allows perfect reception at each subscriber's home as long as the noise signals induced on the digital waveform are not large enough to cause uncorrectable bit errors in the data stream. Similarly, the storage of audio and video information in digital form allows an unlimited number of copies of a movie or program to be made without any loss in quality as compared to the original master recording.

- **Increased number of channels**

Data in digital form can be compressed through digital signal processing techniques, whereas compression on the original analog signal is not possible. The compression rate

achievable with modern audio and video compression algorithms is great enough to allow an increase by a factor of at least ten in the number of programs which can be transmitted over a given bandwidth. This increase in the number of achievable channels can provide enough bandwidth to support wide-area video on demand applications using network bandwidths which are available today.

- **Integrated network solutions**

Transmission of audio and video information in digital form provides a cost savings in that a single integrated network can be used for the delivery of media which has traditionally been transmitted in analog form (audio, video, and telephony), as well as data which is naturally represented in digital form (computer data, control information, etc.). It is likely that digital interactive television networks will support not only video on demand applications, but also such applications as access to the Internet, database lookup, telephone service (both voice and video phone service), home shopping, multi-player video games, and video conferencing. Most of these applications will require a mix of both analog information, such as audio and video, and digital information, such as computer data and control information. Transmission of the analog information in digital form allows a single integrated network to support such interactive multimedia services.

1.2 Enabling Technologies for Interactive Television

Recent advances in several key technologies will allow interactive digital television to become both economically and technologically feasible in the near future. In order to justify the feasibility of wide-area video on demand applications, and thus the need for a transport

protocol capable of transmitting digital audio and video information, this section summarizes these advances.

1.2.1 Audio and video compression standards

When digitized, an analog television signal requires over 160 Megabits per second of storage and transmission bandwidth. (This includes both audio and video data.) This is a fairly large data rate when measured against the bandwidth provided by today's networking standards. Current state-of-the-art wide-area networks are based on the Asynchronous Transmission Mode (ATM) architecture. When transmitted over fiber optic OC-3 links, ATM networks provide a bit rate of 155 Megabits per second (minus some overhead). Thus, a dedicated ATM OC-3 link does not provide enough bandwidth to carry a single digitized television signal.

A city-wide interactive television system must potentially support thousands of simultaneous subscribers, with each subscriber's television program requiring dedicated network bandwidth. Thus, it is clear that data compression must be used in order for wide-area interactive television distribution to be economically feasible. The data compression technique used for interactive television distribution should meet two key requirements. First, the compression algorithm should adhere to an internationally recognized standard. Second, the complexity of the algorithm should not be so great that decompression cannot be performed within VLSI. Conformance to these requirements will allow the industry to benefit from the economics of large scale production and competition among IC vendors offering audio and video decompression integrated circuit components.

Recognizing this fact, the International Standards Organization formed the Motion Picture Experts Group (MPEG). The charter of the MPEG committee was the development of industry standard video and audio compression techniques in which the decompression algorithm could be implemented within VLSI.

The MPEG committee formed two sets of standards which are suitable for interactive television applications. The MPEG-1 standard allows a digitized television signal to be compressed to a data rate of about 1.5 Megabits per second at a quality level comparable to that of VHS video tape (LeGall 1991). The MPEG-2 standard allows a digitized television signal to be compressed to a data rate of 3 – 4 Megabits per second at a quality level comparable to that of the National Television Standards Committee's (NTSC) definition for broadcast television signals. The NTSC definition is used for terrestrial and cable TV broadcast signals in the United States. The impact of the MPEG standards on a digital audio/video transport protocol will be examined in Chapter 2.

1.2.2 Fiber optic networks

Even when digital compression algorithms are used, a video on demand system requires an enormous network bandwidth in order to support the number of expected subscribers in an average size city. Fiber optic network links can provide this bandwidth at a cost which is low enough to make video on demand applications economically feasible. Although fiber optic links are currently too expensive to be used for the connection to each subscriber's home, they can be used to deliver multiple video streams from a video file server at a cable TV head-end office or a telephone company central office to neighborhood distribution nodes. Optical to

electrical converters at the neighborhood distribution nodes can then be used to retransmit the data onto a city's existing cable television distribution plant or onto existing copper twisted-pair telephone lines. Chapter 2 examines the likely ways in which fiber optic and copper based transmission networks will be combined in order to support interactive television, and the impact of these architectures on a transport protocol for digital audio and video transmission.

In addition to providing very large bandwidths, fiber optic transmission lines offer much greater immunity to electromagnetic noise than copper transmission lines. This is important since error correction schemes based on packet retransmissions can not guarantee to meet the real-time delivery deadlines required by digital audio and video data streams.

1.2.3 Digital modulation and transmission techniques

In the United States each NTSC television channel is allocated a 6 MHz bandwidth from the RF spectrum for air-wave or cable broadcast transmission. Current digital modulation methods such as quadrature amplitude modulation (QAM) can encode a bit rate of 30 – 40 Megabits per second within a 6 MHz bandwidth (Quinnell 1994, Chang et al. 1994). When combined with a digital video compression rate of 3 Megabits per second, quadrature amplitude modulation can thus increase the number of programs broadcast in a given bandwidth by a factor of more than 10.

The coaxial cable used for city-wide cable television distribution networks typically has a bandwidth of between 500 and 1000 MHz, allowing between 80 and 160 simultaneous NTSC analog programs to be broadcast. When digital compression and modulation are used, the number of programs that can be transmitted over a single cable TV network is greater than

1000. Chapter 2 discusses how such a network can be used to support interactive television for large metropolitan areas with hundreds of thousands of subscribers.

1.2.4 Disk storage systems

Even with the use of data compression, a video on demand file server requires a substantial amount of disk storage capacity. For example, at a compressed bit rate of 3 Megabits per second, a 2 hour movie requires 2.7 Gigabytes of storage. A typical video on demand file server may be required to store tens or hundreds of such movies.

The cost of this disk storage is often the largest contributing factor to the overall cost of a video server. However, the prices of hard disk drives have decreased rapidly during the past year, while their storage densities have increased just as rapidly. It is now possible to buy magnetic disk drives with storage capacities of over 4 Gigabytes at a cost of less than 40¢ per megabyte. In addition, storage densities on magnetic disk drives are currently increasing at a sustained growth rate of over 60% per year, and this trend is projected to continue past the end of the decade (Ruemmler and Wilkes 1993). As the storage density of disk drives increases at this pace, the cost per megabyte can be expected to decrease at a similar pace.

Furthermore, new "*video friendly*" disk drives have begun to appear on the market. Traditional magnetic disk drives can occasionally suspend the processing of I/O requests in order to perform a thermal recalibration procedure. This procedure can require from 500 to 800 milliseconds, but is otherwise transparent to a processor which issues I/O requests to the drives (Ruemmler and Wilkes 1993). However, this thermal recalibration procedure severely interferes with the real-time retrieval requirements of video on demand applications. A video

friendly disk drive does not interrupt service for this thermal recalibration step, and thus can provide guaranteed access delays for real-time video and audio retrieval and transmission.

In addition to storage capacity, the I/O bandwidth of a disk subsystem has a great impact on the number of simultaneous audio/video streams which a video-on-demand file server can support. A recently developed disk subsystem architecture known as "*Redundant Arrays of Independent Disks*", or RAID, allows multiple disk drives to operate in parallel in order to provide a bandwidth which approaches the aggregate bandwidth of the individual disk drives (Ganger 1994). RAID subsystems have been made commercially available recently, and first generation video on demand file servers are expected to be designed using RAID disk architectures.

1.3 Protocol Requirements for Continuous Media Data Transmission

Of the different media which may exist in an interactive television application, digitized audio and full-motion video place extra demands upon the network. These media forms are only meaningful when the individual samples are played back uninterrupted over a continuous interval of time. Thus, the network transmission protocol must guarantee that the audio and video data will be delivered fast enough so that the receiver always has the next sample available when it has finished playing out the current sample, yet slow enough that the protocol doesn't overflow the receiver's buffer space. Such media forms are referred to as "*continuous media*". As explained below, the quality of service offered by conventional data communications protocols does not adequately support the requirements of continuous media data transmission.

1.3.1 Requirement for intramedia synchronization

Conventional data communications protocols have been designed to provide error-free data delivery through the use of error control coding and retransmission. A side effect of such protocols is that packet delivery times in the network are unbounded, due in part to the unpredictable number of retransmissions that a packet may suffer. Nevertheless, such error-free data transmission has been a requirement for most networking applications to date, while fast data delivery has been desirable but not essential.

For transmission and playout of continuous media data streams, the tradeoffs between error-free transmission and bounded transmission delays are reversed. That is, continuous media applications require guaranteed bounds on data delivery times in order to prevent buffer underrun or overflow during playout at the receiver. However, these applications can tolerate occasional bit errors or packet losses since such errors typically introduce only small distortions in an audio or video sample. These distortions are momentary, and may not even be perceived by the viewer. Thus, while conventional communication protocols provide guaranteed reliability at the expense of undeterministic delay, protocols for continuous media transmission require bounded transmission delays at the expense of occasional bit errors. Protocols which can meet such real-time delivery constraints for continuous media data streams are said to provide *intramedia* synchronization.

1.3.2 Requirement for ntermedia synchronization

Video on demand applications may require the transmission and playback of multiple related continuous media streams in a synchronized fashion. For example, a video stream will

often require an associated audio stream to be transmitted simultaneously. If the audio is to be played back in stereo, then transmission of two audio streams will be required. At the receiver, the play-back of the audio and video streams must occur in a synchronized fashion.

It may be possible to multiplex the data for the different media streams into a single continuous media file. If this is done, the server needs only to transmit a single file to each client. However, this is not necessarily the optimal choice. For example, an audio narration may be recorded and stored separately from the video which it describes. In fact, several such audio narration files may be recorded, each in a different language. In this case the audio file which is to be transmitted can be selected based on the language preference of the viewer.

If the audio and video data are stored and transmitted as separate streams, a continuous media transport protocol should allow for these different media streams to be played back in a synchronized manner at the receiver. As an example, voice and video must remain closely synchronized during playout to prevent the familiar lip-sync problem which is sometimes observed when watching a television broadcast for which the audio and video synchronization has been lost. A protocol which supports the simultaneous transmission and synchronized playback of several continuous media files is said to provide *intermedia* synchronization.

The research addressed by this dissertation is the development of a continuous media transport protocol which supports both intramedia and intermedia synchronization. The continuous media transport protocol is developed specifically to support the requirements of wide-area video on demand distribution.

1.4 Related Work

This section discusses the existing research which is most relevant to the work presented here.

(Ramanathan and Rangan 1993) describe feedback mechanisms for intramedia synchronization in networks with bounded delivery times and without a global clock source. However, unlike the research proposed here, their mechanisms work only for fixed bit-rate compression techniques, and do not work in the more general situation where variable bit-rate compression is used. In addition, their approach ignores the network transmission time of each packet, thus assuming an infinite network bandwidth is available. The protocol developed in this dissertation accounts for the finite network bandwidth of real networks.

(Ramanathan and Rangan 1993) also proposes an intermedia synchronization mechanism assuming the audio and video streams must be resynchronized at the server. That is, they assume that the modules responsible for video and audio playback at the client cannot communicate directly with each other.

(Anderson and Homsy 1991) propose an intermedia synchronization mechanism for multiple continuous media streams whereby the client skips or replays samples of one stream in order to keep it synchronized to another (master) stream. They do not consider the consequences of their proposed mechanism on intramedia synchronization. The protocol developed here includes the effects of intermedia synchronization on intramedia synchronization.

(Wolfinger and Moran 1991) proposes a set of transport layer service primitives to

allow the coordination of multiple data streams being played out as part of the same multimedia application. Their work does not incorporate these primitives into an actual transport protocol.

(Little and Ghafoor 1990) and (Little and Ghafoor 1991) propose the use of Petri Net techniques to model the requirements of multimedia synchronization. Their work is mainly relevant to event-driven synchronization, such as synchronizing a text overlay with a slide presentation. They do not propose techniques for continuous media synchronization.

To the author's knowledge, no methods have been proposed to date which incorporate both intra- and inter- media synchronization while realistic network architectures. The protocol developed in this dissertation addresses these issues.

1.5 Organization of this Dissertation

The remainder of this dissertation is organized as follows:

Chapter 2 examines the overall system architectures likely to be used for wide-area video on demand systems. From this study, a set of protocol requirements for the protocol developed in this dissertation is formulated.

Chapter 3 presents a continuous media transport protocol which meets the real-time requirements for video on demand applications. A key feature of this protocol is the calculation of a data transmission schedule which guarantees to meet the real-time delivery deadlines for continuous media data streams without overflowing the buffer space at the receiver. It is shown in Chapter 3 that the transmission schedule used by the continuous media transport protocol developed in this dissertation is one which is able to meet the play-out

calculation of a data transmission schedule which guarantees to meet the real-time delivery deadlines for continuous media data streams without overflowing the buffer space at the receiver. It is shown in Chapter 3 that the transmission schedule used by the continuous media transport protocol developed in this dissertation is one which is able to meet the play-out deadlines at the receiver without overflowing the receiver's buffer space, while minimizing the buffer requirements at the receiver. (i.e., It is shown that it is not possible to compute a transmission schedule with a smaller buffer requirement at the receiver and still guarantee to meet the delivery deadlines at the receiver.)

The version of the continuous media transport protocol presented in Chapter 3 is designed to operate over ideal networks. Thus, it ignores several important properties of practical networks. In Chapter 4, the protocol developed in Chapter 3 is extended to compensate for the non-ideal properties of practical networks.

Finally, Chapter 5 summarizes the research contributions of this dissertation, and proposes several possible related topics for future study.

CHAPTER 2. VIDEO ON DEMAND PROTOCOL REQUIREMENTS

This chapter examines the system components and the end-to-end network architectures most likely to be used for large scale video on demand applications. From this study, a list of requirements is formed for a continuous media transport protocol. A protocol which meets these requirements is presented in Chapters 3 and 4.

2.1 Assumed Environment

Figure 2.1 illustrates the environment under which the protocol developed in this dissertation is assumed to operate. A video on demand continuous media file server stores pre-recorded and pre-compressed audio and video files. Users can interact with the system to select a program or movie of interest. The selection process is possibly performed through on-screen menu information which is downloaded from the file server, with user selection via a remote control device. The menu download and selection processes will typically require a reliable, non-real-time transport protocol and are therefore assumed to happen outside of the continuous media transport protocol presented in this dissertation.

Once a movie selection has been made, a message identifying the selected program is transmitted across the network to the continuous media file server. This message is transmitted over the reliable, non-real-time connection which was established for the initial menu download. The continuous media file server will then attempt to set up one or more real-time continuous media transport connections as required for the requested movie. If these

Video on Demand File Server

- Stores Continuous Media Files
- Variable Bit Rate Compressed Streams
- Multiple Related Files per Movie

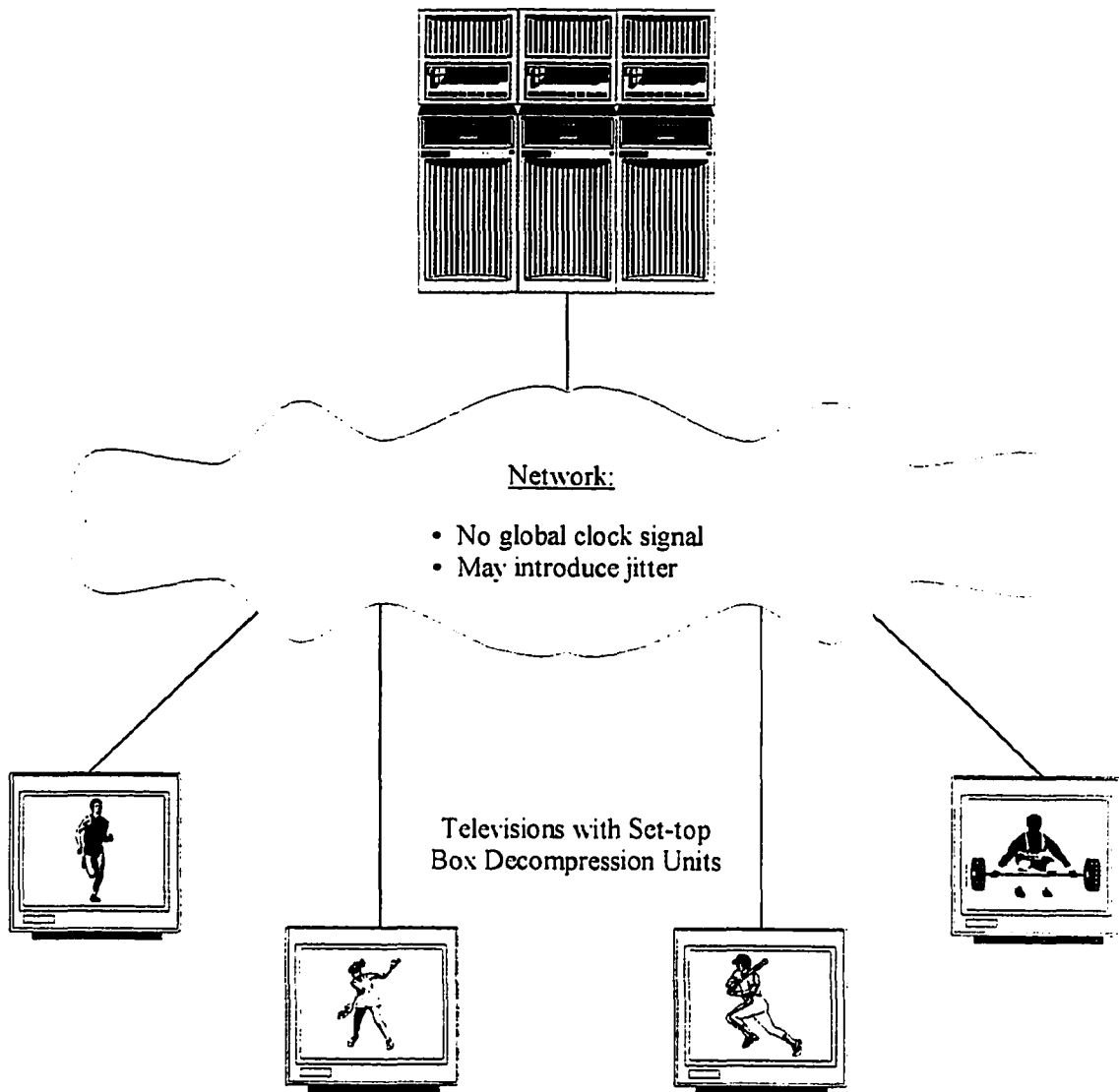


Figure 2.1. Operating Environment for the Continuous Media Transport Protocol

connections are successfully established, the server will then begin transmission of the selected movie over these connections, using the continuous media transport protocol developed in this dissertation.

The transmitted data is received and buffered in a “*set-top box*” located at the subscriber’s premises. Hardware in the set-top box removes the compressed audio and video samples from the buffer, decompresses them, and plays them back in real-time.

The different media streams (such as audio and video tracks) which constitute a program are assumed to be stored as separate files. A database stored on the continuous media file server contains references to the individual media files which form a program. Each continuous media file which is part of a program is transmitted across the network over a separate transport layer connection. Transmission is such that play-out of the different continuous media files occurs in a synchronized manner at a client.

In addition to references to the individual files, the server’s database is assumed to contain information which specifies the play-out deadlines for the individual audio samples or video frames in each continuous media file. The format of this play-out deadline information, and a description of its use is given in Chapter 3.

The key features of this environment are described below.

2.1.1 Transmission of stored multimedia data only

In general, distributed multimedia applications can be classified as operating with either live data or stored data. In applications which operate with live data, the data is generated (digitized) in real-time during the execution of the application program. Typical examples

include video conferencing, remote expert consultation, joint editing, and multi-user game playing. Distributed applications which operate on stored data access pre-recorded data from a remote multimedia file server. Examples include play-back of digitized movies for entertainment purposes, viewing of multimedia educational or training programs, and access to multimedia catalogs for computerized shopping. The continuous media transport protocol developed in this dissertation is designed to operate with stored multimedia data only.

The reason the protocol developed in this dissertation is restricted to the transfer of stored data is that a single transport protocol to transmit both live and stored continuous media data streams may not be the optimal choice. A protocol transmitting stored data has more flexibility in scheduling the times at which data is to be transmitted. For example, since data is not generated in real-time it can be pre-fetched and transmitted in bursts to the receiver. In general, with stored multimedia data the continuous media server and protocol can take advantage of the knowledge of future data transmission requirements in order to optimize the data transfer. The transport protocol developed in this dissertation does take advantage of such information to calculate a transmission schedule which minimizes buffer requirements at the receiver. This optimization is discussed further in Chapter 3.

2.1.2 Bounded transmission delays

The ability to meet the real-time requirements of continuous media transmission may be provided by the network on a guaranteed basis, or may be attempted by the network on a best-effort basis. In order for a transport protocol to provide guaranteed real-time service to an application, the underlying protocol layers (physical, data link, and network) must also

guarantee real-time service to the transport layer. Much recent research has focused on the design of routing, queueing, and admission control policies to guarantee such service (Ferrari and Verma 1990), (Ferrari 1992), (Kurose 1993).

Real-time guarantees from the underlying protocol layers can be achieved in one of two ways. If the underlying network is a local area network for which the worst-case media access time is guaranteed (such as FDDI), the transmission delay bounds can be determined from knowledge of the packet transmission rate required for each stream and the bounded media access delay. If the underlying network is a multi-hop network, an end-node must declare its worst case traffic parameters (e.g. maximum packet size and minimum time interval between successive packets) as well as its real-time quality of service requirements at connection establishment time. Admission control policies are then invoked by each packet switch during the connection establishment phase, and the connection is accepted only if the real-time quality of service requirements can be met. Networks based on ATM protocols allow such quality of service negotiation.

The protocol developed in this dissertation assumes that the underlying protocol layers do provide some form of real-time support in terms of guaranteed delay bounds for each packet transmitted across a connection. The protocol uses these delay bounds as well as knowledge of the amount of buffer space reserved by the receiver for incoming packets to guarantee that continuous media data streams are transmitted such that the receiver never suffers from data overrun or underrun during play-out.

The protocol developed here may also work in a network which provides statistical,

rather than deterministic, delivery bounds. In this case, however, the protocol's ability to guarantee that data is delivered by its play-out deadline becomes a statistical, rather than absolute, guarantee.

In addition to real-time guarantees from the network, real-time scheduling is required from both the multimedia server (in the form of disk scheduling and scheduling of data transmission requests to the network), as well as the operating system and hardware at the receiver (for scheduling of decompression and play-out). Although the actual real-time scheduling mechanisms of the multimedia server and the client are outside the scope of this protocol, it is assumed that such mechanisms do exist.

2.1.3 Necessity for intermedia synchronization

As noted in Chapter 1, multiple continuous media streams may coexist to form a single continuous media program. The play-back of these streams must be synchronized to within certain limits. For example, research has shown that the maximum unperceptible skew which may occur during play-out of simultaneous audio and video tracks is about 100 msec (Anderson and Homsy 91). If the skew exceeds this, then the audio and video appear to have lost lip synchronization. To prevent this, some form of intermedia synchronization is necessary. Therefore, the protocol developed in this dissertation supports the transmission and synchronized play-back of one or more continuous media streams.

2.2 Protocol Requirements Due to Network Architecture

Large scale video on demand systems, for example those spanning a sizable

metropolitan area, will require an enormous networking infrastructure in order to reach the perhaps tens or hundreds of thousands of homes in their serving area. Implementing such a network from the ground up would require a huge capital investment, thus lessening the economic viability of such systems. Therefore, an existing metropolitan area network that reaches most, if not all of, the homes in its serving area should be used if possible.

In most cities, two networks are currently in place which are capable of supporting video on demand applications: cable television distribution networks and the telephone switching network. Cable television networks currently pass by more than 90% of the homes in the United States. The percentage of homes with access to telephone service is even higher. Therefore, initial interactive television systems will likely be based on one of these two network architectures. This section examines the feasibility of these two types of networks for video on demand applications, and the impact of these networks on a transport protocol designed for continuous media data transmission.

2.2.1 Video On Demand Over Cable Television Distribution Networks

Figure 2.2 (Miller 1994) illustrates the network architecture of a typical first generation cable TV system. Satellite and terrestrial broadcast antennas at the cable TV head-end office receive television signals from remote locations. In addition, signals for locally generated programs and commercials may originate directly at the cable TV head-end office. The signal for each television program is modulated onto a dedicated RF carrier channel and transmitted onto the cable television distribution network using frequency division multiplexing. The distribution network consists of segments of coaxial cable connected in a tree and branch

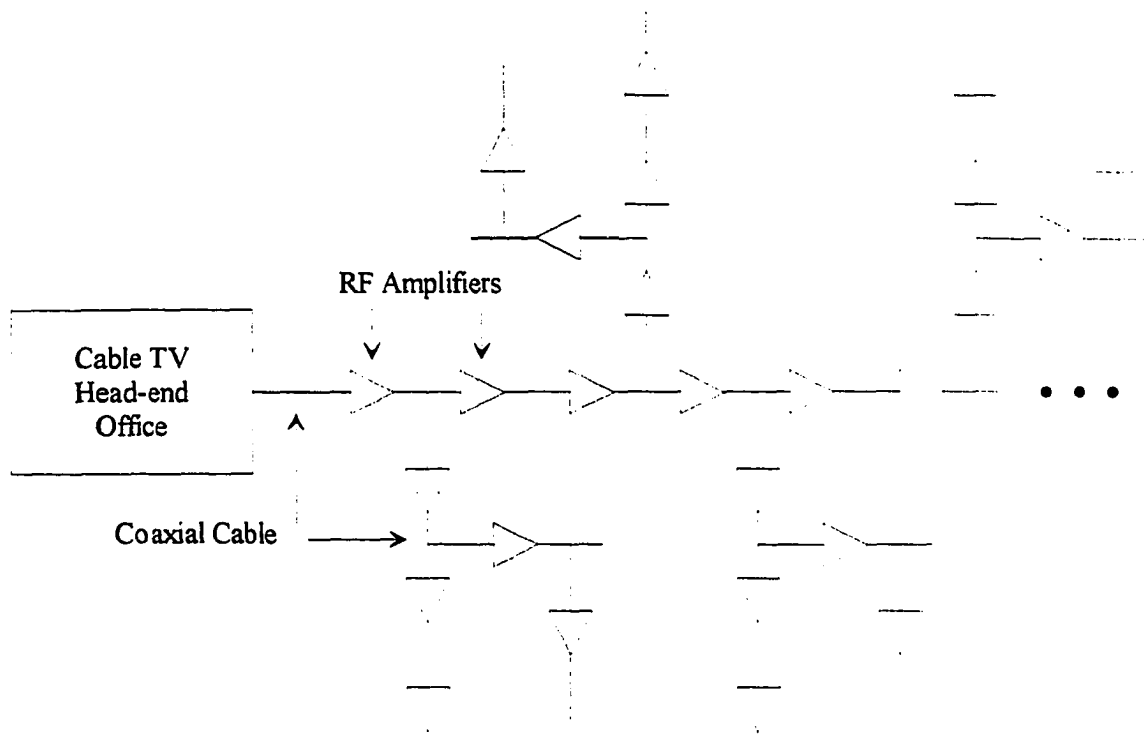


Figure 2.2. First Generation Cable Television Distribution Network

topology. Amplifiers are spaced periodically throughout the network in order to compensate for signal losses in the cable.

A typical coaxial cable distribution network using the architecture shown in Figure 2.2 provides an end-to-end bandwidth of approximately 450 – 500 MHz. Each individual television channel is assigned a bandwidth of 6 MHz from this spectrum. Thus, such a system allows a maximum of approximately 80 television channels. For large scale (city-wide) video

on demand applications this is clearly insufficient since such a system can honor a maximum of only 80 simultaneous movie requests at any one time.

The bandwidth limitation of the coaxial cable distribution network is primarily due to two effects: First, each amplifier in the signal path adds some amount of distortion and noise, thus degrading the original signal. In a typical system, a television signal will pass through 30 – 40 of these amplifiers before reaching a subscriber (Miller 1994). Second, electromagnetic interference may be induced in the cable as it passes near broadcast antennas and power substations, further limiting the bandwidth.

Current generation cable distribution networks have replaced the coaxial cable trunk lines in Figure 2.1 with fiber optic lines. Fiber optic lines have much greater immunity to signal loss and induced interference than coaxial cable. By using fiber optic lines to traverse the majority of the distance from the cable TV head-end office to a subscriber's home, the coaxial cable segments can be shortened. Thus, fewer RF amplifiers are needed in the signal path and less electromagnetic interference is induced into the cable. Such architectures can increase the end-to-end bandwidth of the network to approximately 1 GHz. The term "*Hybrid Fiber Coax*" is used to describe such networks.

When combined with digital video compression and state-of-the-art modulation methods for digital communications, hybrid fiber coax networks provide sufficient bandwidth for interactive television applications.

When hybrid fiber coax architecture are adopted for interactive television applications, the video servers will likely be attached to a high-speed fiber optic ATM switching network.

The ATM network will route data for each video stream to an optical-to-electrical neighborhood distribution node. In order to support a large population of users, there may be several such video servers in a system. The neighborhood distribution nodes will modulate the digital data received from the ATM network onto a 6 MHz carrier wave using a digital modulation technique such as quadrature amplitude modulation, and transmit the resulting signal onto the neighborhood's coaxial cable television network.

The choice of 6 MHz RF channels in a hybrid fiber coax network maintains compatibility with existing cable television modulation equipment. In fact, the architecture described above allows the same network to carry analog NTSC channels on some of the 6 MHz frequency bands, and compressed digital programs on the others. Thus, some homes may choose to subscribe to only the existing analog based cable television service, while others can choose to subscribe to the digital interactive television service.

A 6 MHz channel using state-of-the-art digital modulation techniques can transmit data at a rate of 30 – 40 Megabits per second. This transmission rate is large enough to supply 10 to 12 homes with MPEG compressed digital video at NTSC quality in the same bandwidth that is required to transmit a single analog channel. Thus, the use of digital compression and transmission allows an increase by a factor of at least ten in the number of programs which can be transmitted. This increase in the number of programs can be enough to allow each active viewer to select and receive a separate program or movie, thus supporting true video on demand applications.

As an example, consider a hybrid fiber coax network with an end-to-end bandwidth of 1

GHz. A typical RF spectrum assignment may be as follows:

- 0 – 50 MHz: Data from subscribers to video server (for movie selection, etc.)
- 50 – 500 MHz: Existing analog television channel transmission
- 500 – 1000 MHz: Digital video on demand channel transmission

With such a bandwidth assignment, there are 83 6 MHz channels available for digital video transmission. Since each channel can carry between 10 and 12 compressed video programs, a neighborhood distribution node can support about 900 active viewers. Assuming that the system is designed to support a peak usage of 30% of its subscribers, each neighborhood distribution node can serve 3000 homes. A city with 150,000 homes (perhaps 300,000 - 400,000 residents) would require 50 neighborhood distribution nodes.

The network architecture described above is representative of architectures being proposed for cable television based interactive television field trials in several U.S. cities. Note that in this architecture, ATM is likely to be used as the transmission network from the video server to the neighborhood distribution nodes. The choice of ATM as the transmission network at the video server and a cable TV distribution network at the subscriber's premises imposes several requirements for a continuous media transport protocol designed for video on demand applications. These requirements are listed and explained below.

- **Support for periodic synchronization between the video server and its clients**

ATM networks transmit data in small, fixed size packets called "*cells*". Each cell may traverse several hops along the route from the video server to the home. In such networks, it

is not feasible for both the transmitter (video server) and receiver (set-top box) to be synchronized to a global network clock signal. Instead, each operates with its own local clock, and the frequency of these clocks will not be perfectly matched. Unless corrective action is periodically taken, the transmitter and receiver will tend to drift out of synchronization over time.

As an example, consider that the client plays back video frames at a nominal rate of 30 frames per second (a rate compatible with current US television standards). The video decompression hardware at the client's set-top box uses a local oscillator to create this timing, and generates an interrupt during each vertical retrace time of the CRT in order that the next video frame may be loaded into video memory. Assume that the oscillator in the set-top box is accurate to 100 parts per million. (This is a reasonable value for such oscillators.) Thus the actual play-back period, T , for a video frame at the client may be in the range:

$$0.9999 \times \frac{1}{30} \text{ sec} \leq T \leq 1.0001 \times \frac{1}{30} \text{ sec}$$

For continuous media play-out, the server, knowing these limits, must transmit data at a rate fast enough such that even if the client's clock is running at its fastest possible rate (corresponding to the smallest period), the next video frame is always available for decompression and play-out at the receiver when play-out of the current frame is complete. However, if the client's clock is actually running at its slowest possible rate, then over time the receiver's buffer space will tend to fill. For example, if the server and set-top box each have a clock tolerance of 100 parts per million and a 2 hour digitized movie compressed to a

bit rate of 4 Megabits per second is to be played back, the server may be leading the client by 720,000 bytes at the end of the movie.¹ Unless corrective action is taken periodically to resynchronize the transmitter and receiver, this accumulated error may cause the receiver's buffer space to overflow.

The protocol presented in Chapters 3 and 4 includes a feedback mechanism by which the set-top box may periodically report the number of bytes by which it and server have drifted from synchronization. The server will use this feedback message to adjust the transmission schedule of future data. Guarantees are provided such that this feedback mechanism will prevent data underrun or overrun at the receiver.

- **Support for real-time guarantees in the presence of jitter**

The term "*jitter*" is defined as the difference between the maximum and minimum delay experienced by a packet in computer network. Since ATM is a packet switched network, the end-to-end transmission delay for a cell depends on the queueing times experienced by the cell at each packet switch. Thus, the jitter for an ATM network may be non-zero. Therefore, the continuous media transport protocol developed in Chapters 3 and 4 must be capable of operating correctly over networks which may potentially introduce jitter into the network transmission delays.

Jitter affects a continuous media transport protocol in two ways: first, the presence of

¹This is calculated as follows: For a clock stability of 100 parts per million, the error between the fastest possible clock and the slowest possible clock may actually be 200 parts per million. At a data rate of 4 Megabits per second, this represents a drift of 800 bits, or 100 bytes, per second. Thus, at the end of the 2 hour movie (7200 seconds), it is possible for the transmitter to be 720,000 bytes ahead of the receiver.

jitter must be taken into account when calculating a transmission schedule that guarantees to meet the delivery deadlines for a continuous media data stream; and second, the possibility of jitter causes some potential techniques for the correction of clock frequency errors to be impractical.

- **Support for Dynamic Bandwidth Assignment**

ATM networks support a dynamic bandwidth assignment. The transmission bandwidth is negotiated with the network switches at connection set up time. The network switches can provide a guaranteed quality of service by reserving resources based on the negotiated bandwidth. However, a higher bandwidth connection requires more resources to be allocated, limiting the number of simultaneous connections which can be supported.

For a continuous media transport protocol operating over networks with a flexible bandwidth assignment such as ATM, the selection of an appropriate value for the bandwidth is important. If too large a bandwidth is negotiated, then network resources are wasted needlessly. If too small a bandwidth is negotiated, then the network may not be able to transmit data fast enough to meet the play-out deadlines at the receiver. The protocol developed in Chapters 3 and 4 can operate over networks with a fixed bandwidth assignment, such as FDDI or T1 links, or over networks with a flexible bandwidth assignment, such as ATM. When the network architecture supports a flexible bandwidth assignment, the protocol computes the minimum bandwidth necessary to meet the play-out deadlines for a specified buffer size at the receiver. This minimum bandwidth is then used for scheduling the transmission of the individual audio or video frames in the continuous media file.

- **Support for Data Multiplexing**

As noted above, a hybrid fiber coax network typically divides the RF spectrum of the coaxial cable into 6 MHz channels. When digital compression and transmission are used, each channel provides enough bandwidth to transmit multiple video programs. At the receiving end, the set-top box passes the incoming waveform through a bandpass filter tuned to the proper 6 MHz channel and demodulates the resulting signal to recover a digital data stream. However, this digital data stream will typically contain data for multiple programs. Thus, a further demultiplexing operation is necessary in order to recover the video and audio streams for the specific program of interest. Thus, a continuous media transport protocol for video on demand applications must provide a mechanism to support this demultiplexing operation.

The protocol proposed in Chapters 3 and 4 of this dissertation transmits data in packets. Each packet is assumed to be prefixed with a header which contains an identifier field. The identifier uniquely identifies the packet as belonging to a particular audio or video track of a specific program. The set-top box can use this packet identifier to select the audio and video tracks for the program of interest.

It should be noted that the header described above is a transport layer header, and is different from the ATM cell header which is appended by the data link layer. The ATM cell headers do contain Virtual Circuit Identifier (VCI) and Virtual Path Identifier (VPI) fields which allow the ATM cell switches to demultiplex and route incoming cells to the proper destination. However, using these fields to demultiplex audio and video packets at the set-top box would require that the ATM cell headers be transmitted onto the coaxial cable distribution

network. Such an approach is fairly inefficient since the ATM cell header imposes an overhead of approximately 10 percent. The approach proposed here is a more efficient solution. Note, however, that this protocol requires that the neighborhood distribution nodes which terminate the ATM distribution network assemble the cells for a complete continuous media packet before transmitting this packet on to the cable television distribution network.

The above discussion can be summarized as follows: A protocol for transmission of video on demand files over ATM networks in a hybrid fiber coax architecture must be designed to provide support for the following:

1. Support for periodic clock synchronization between the video server and set-top boxes
2. Support for real-time guarantees in the presence of jitter
3. Support for dynamic bandwidth assignment
4. Support for data multiplexing

The protocol described in Chapters 3 and 4 is able to meet these requirements.

2.2.2 Video On Demand Over Telephone Switching Networks

Using digital modulation techniques, ordinary copper twisted pair telephone lines can be used to carry digital data at rates which may be sufficient for compressed video applications. A recently developed modulation scheme known as "*Asymmetric Digital Subscriber Line*" (ADSL) can transmit data downstream (from the video server to the home) at a rate of 1.5 Megabits per second with an upstream channel of 16 Kilobits per second at the same time that the line is carrying a normal analog telephone call (Chang et al. 1994). This is sufficient to carry an MPEG-1 compressed program with VHS video tape quality. A newer technology,

known as ADSL II, increases the downstream transmission rate to 6.4 Megabits per second. This bandwidth is more than sufficient to carry an MPEG-2 compressed program at NTSC quality.

A video on demand system using ADSL or ADSL II technology can be constructed in two possible architectures. The first architecture is similar to the hybrid fiber coax network architecture used for cable television based video on demand systems. In this architecture, a video server transmits data across an ATM network to ADSL modems. Each ADSL modem modulates the data it receives from the ATM network onto a twisted pair telephone line. For such an architecture, the restrictions imposed on the transport protocol are the same as those defined for the case of cable television distribution networks with the exception that support for data demultiplexing at the set-top box is not a requirement.

The second possible architecture for video on demand transmission over telephone networks is for the video server to be directly connected to the ADSL modems. In this case, the requirement of support for dynamic bandwidth assignment is also eliminated. This second architecture generalizes to the case of a video server which transmits each program over a dedicated link, with each link having a fixed bandwidth.

The protocol developed in this dissertation supports continuous media transmission over both networks which have a flexible transmission rate, such as ATM, and networks which have a fixed transmission rate such as ADSL or T1 lines. (Quinnell 1994) describes an example of a system which requires continuous media transmission over T1 lines. In this system a San Francisco television station uses T1 links to transmit digitally encoded and

compressed local television shows to cable TV head-end offices in remote cities. MPEG-1 is used as the compression format. .

2.3 Protocol Requirements Due to Compression Algorithms

This section examines current state-of-the-art digital audio and video compression standards and formulates a list of protocol requirements that the use of compression imposes on the transport protocol developed in this dissertation.

Although the protocol developed here is fairly independent of the specific compression algorithm used, the MPEG-1 standard is examined as an example of a typical compression algorithm used for compression of continuous media audio and video data. The protocol requirements due to the use of MPEG-1 compression are representative of the requirements which exist for the transmission of compressed audio/video data in general.

2.3.1 Requirement for compression

An uncompressed full-motion digitized video signal requires a tremendous amount of storage capacity for even a short duration sample. Correspondingly, a tremendous amount of bandwidth is required to transmit the digitized data across a network in order for it to be played back at the receiver in real-time. These storage capacity and bandwidth requirements are estimated below.

The NTSC television broadcast standard used in the United States assigns a 6 MHz bandwidth from the RF spectrum to each television channel. This signal encodes both the audio and video information, although the video signal occupies most of the channel's

bandwidth.

For compression purposes, the video signal is typically decomposed into three components: a luminance component and two chrominance components. The luminance component encodes the brightness information and the two chrominance components encode the color information.

According to Shannon's Sampling Theorem, when an analog signal is digitized it must be sampled at a rate of at least twice that of its highest frequency component to avoid anti-aliasing. Before sampling, the analog signal must first be passed through an anti-aliasing filter to remove frequency components greater than half the sampling rate. In practice the sampling rate is usually greater than twice the filter's cutoff frequency since the anti-aliasing filter cannot be designed to behave as a perfect (brick-wall) low pass filter.

Thus, according to Shannon's Sampling Theorem, the three components of a 6 MHz NTSC video signal must each be sampled at a rate greater than 12 MHz. In practice, however, the human eye cannot easily perceive high frequency changes in the chrominance components. Therefore most professional digital video equipment samples the luminance component of an NTSC video signal at 13.5 MHz with an 8-bit resolution, and the two chrominance components at 6.75 MHz each, also with an 8-bit resolution. The uncompressed bandwidth, B , required for transmitting such a digitized video signal can thus be calculated as follows:

$$\begin{aligned} B &= 13.5 \times 10^6 \text{ samples / sec} \times 8 \text{ bits / sample} + 2 \times 6.75 \times 10^6 \text{ samples / sec} \times 8 \text{ bits / sample} \\ &= 216 \times 10^6 \text{ bits / sec} \end{aligned}$$

It is possible to reduce this bandwidth to about 162 Megabits per second by eliminating sampling during the horizontal and vertical retrace periods of the signal. However, even at this reduced rate, a one hour uncompressed movie requires over 97 Gigabytes of storage, and an individual uncompressed video stream requires 162 Megabits per second of network bandwidth.

2.3.2 The MPEG compression standard

As discussed above, for video on demand applications to be feasible with storage media and network technology which are available today, it is apparent that data compression must be used. The International Standards Organization (ISO) has formed a committee known as the Motion Picture Experts Group (MPEG) to recommend standards for the compression of full-motion digitized video data and its associated audio. The MPEG-1 committee's standards are organized into three components: MPEG-1 Video, MPEG-1 Audio, and MPEG-1 System.

MPEG-1 Video

The MPEG-1 Video standard addresses the compression of digital video to rates of about 1.25 Megabits per second. The video compression process produces three distinct types of compressed video frames:

- **Intra-coded frames (I-frames)**

I-frames are compressed using only information from the frame being compressed. (i.e., An I-frame is encoded as though it is a still image.) No information from surrounding frames is used to compress an I-frame, and therefore no compression to remove the redundancy

between frames is performed. I-frames serve as access points for random accesses into a sequence of video frames.

- **Predictive-coded frames (P-frames)**

P-frames are compressed using motion compensation techniques to remove temporal redundancy between the frame being compressed and a previous frame. The motion compensation algorithm divides the frame being compressed into regions of 16×16 pixels. For each such region, a difference matrix is calculated which records the difference between the pixels in the region being compressed and a “similar” region in the most recently compressed I- or P-frame. Since the difference values between two similar regions is usually small, the number of bits required to store these difference values is also small.

- **Bi-directional Predictive-coded frames (B-frames)**

B-frames are compressed using motion compensation techniques similar to those used to compress P-frames. However, for B-frames the difference matrix represents the difference between the actual pixels in a region and a matrix of interpolated values from a region of both a past and a future I- or P-frame.

Compression of a B-frame requires that a subsequent I- or P-frame in the video stream be already digitized in order to serve as a reference frame. Thus, a look-ahead procedure is required by the compression hardware and software. Similarly, decompression of a B-frame requires that the subsequent frame to which it is referenced be already decompressed. To support this requirement, the MPEG compression algorithm stores and transmits frames in the order in which they should be decompressed. When B-frames are present, this order will differ

from the order in which the frames should be displayed.

The MPEG-1 video compression algorithm imposes two requirements on a continuous media transport protocol:

1. At discrete points in time, the receiver will remove the data for a complete compressed frame from its buffer, decompress it, and play it back. Therefore, the delivery deadlines at the receiver can be represented as a finite list of discrete times at which a specific amount of data, corresponding to the compressed frame size, must be available in the receiver's buffer. A continuous media transport protocol must guarantee that these delivery deadlines are met.
2. The size of each compressed frame can vary widely. A typical size for an I-frame is about 30 K bytes for NTSC quality video. However, an I-frame is typically about twice as large as a P-frame and 5 – 10 times as large as a B-frame. Thus, the amount of data which must be present in the receiver's buffer by the play-out deadline for each frame varies with each frame. Therefore, a continuous media transport protocol should be designed to operate with video frames of varying sizes.

MPEG-1 Audio

The MPEG-1 Audio standard addresses the compression of digital audio to rates of 64 Kilobits per second, 128 Kilobits per second or 192 Kilobits per second. The compressed audio stream has similar characteristics to that of a compressed video stream. That is, the compressed audio stream can be modeled as a sequence of variable size frames, each typically several kilobytes in size. At discrete points in time, the audio decompression hardware

removes the next audio frame from its buffer to play it back.

MPEG-1 System

The MPEG-1 System standard addresses the issues of multiplexing related audio and video streams into a single stream and time-stamping the data samples in the streams so that they can be demultiplexed and played out in a synchronized manner by the receiver.

The MPEG-1 System standard addresses some of the same issues as the protocol developed in this dissertation. However, the transport protocol developed here improves on the MPEG-1 System standard in several ways. First, the MPEG-1 System standard does not adequately address the problem of correcting for clock frequency errors in the presence of network jitter, whereas the protocol developed in this dissertation does. Second, the protocol developed in this dissertation allows both constant bit rate and variable bit rate compression to be used. Furthermore, the protocol developed here guarantees that the transmission schedule used by the server to transmit a continuous media stream is one which minimizes the buffer space at the receiver. The MPEG-1 System standard does provide such a guarantee.

2.4 Statement of Protocol Goals

The research described in this dissertation is the development of a continuous media transport-layer protocol for stored continuous media data which provides for intra- and inter-media synchronization. The protocol uses transmission delay guarantees from the underlying network in order to provide intramedia synchronization. The protocol operates in networks where a global clock signal is not available by using feedback messages from the receiver to

dynamically resynchronize with the server. Knowledge of the receiver's buffer space, as well as the accuracy of the receiver's clock is used to determine when resynchronization is necessary. A significant contribution of the research presented in this dissertation is the ability to provide this resynchronization for variable bit-rate compressed data, as existing proposals only provide this synchronization for fixed size data frames (Ramanathan and Rangan 1993).

The transport protocol here is designed to use the guaranteed transmission delay bounds of underlying protocol layers in order to meet the real-time requirements of multiple continuous media streams. Finally, since the different media streams involved in a continuous media program may be transmitted over several simultaneous transport-layer connections (as opposed to multiplexing them across a single connection), the proposed protocol includes mechanisms to maintain synchronization of the different media streams during play-back.

CHAPTER 3. THE CONTINUOUS MEDIA TRANSPORT PROTOCOL

This chapter describes the design of a transport protocol which meets the real-time delivery deadlines for continuous media data streams without overflowing the allotted buffer space at the receiver. The protocol developed here is named the *Continuous Media Transport Protocol* (CMTP). In this chapter a version of CMTP is designed which operates correctly in a simplified, ideal environment. In Chapter 4 the protocol developed here is refined to operate correctly in realistic, non-ideal environments.

3.1 Protocol Operating Environment

The version of CMTP described in this chapter requires several simplifying assumptions regarding the network environment under which the protocol can operate. These assumptions allow the basic operation of the protocol to be described and understood. In Chapter 4 the version of CMTP developed in this chapter is refined in order that these assumptions can be removed.

The simplifying assumptions made in this chapter are as follows:

1. **Errors due to non-synchronized clocks do not occur.**

The continuous media server and all of its clients are assumed to be frequency locked to the same clock source. Thus, the gradual overrun or underrun of data at a receiver's buffer due to slight differences in the clock frequencies used by the transmitter and receiver is avoided. This assumption is removed in Chapter 4.

2. The network transmission delay jitter is zero.

Every packet transmitted through the network experiences the same end-to-end transmission delay. This assumption is valid for single-hop networks for which a separate physical link is dedicated to each media stream at the server. However, for networks with multiple hops, the non-deterministic queuing delay at each packet switch can result in an end-to-end delay which is variable. Likewise, in networks with a shared transmission medium, the non-deterministic media access delay can also result in an end-to-end delay which is variable.

In Chapter 4 this restriction is loosened to allow a variable transmission delay. However, in this case the end-to-end delay jitter, defined as the difference between the maximum and minimum end-to-end delay, is required to be bounded.

3. Intermedia synchronization is not necessary.

The protocol developed in this chapter does not provide any mechanisms for synchronizing the play-back of two continuous media streams at a client. Thus, a client may play back a video stream without audio, or an audio stream without video. However, play-back of separate audio and video streams in a synchronized manner is not supported. This restriction is removed in Chapter 4.

In addition to the temporary restrictions stated above, the following additional restrictions are placed upon the system. These restrictions are realistic for practical video-on-demand environments, and therefore they will not be removed by the protocol enhancements in Chapter 4.

1. Data is transmitted by the server in the order in which it is to be decompressed.

Audio and video frames are to be transmitted in the order in which they are to be decompressed. If this order differs from the order in which the frames are to be displayed (as, for example, with MPEG video compression when B-Frames are present), the frames which have been decompressed but are to be delayed before being displayed are stored in a *reorder buffer* at the receiver. For calculation of buffer space requirements at the receiver, this reorder buffer is assumed to be separate from the receive buffer used by the Continuous Media Transport Protocol.

2. The underlying network provides in-order data delivery.

Data is delivered by the network in the same order in which it is transmitted. This assumption is valid for connection oriented (i.e., “virtual circuit”) packet switched networks such as ATM. However, the protocol developed here is not designed to operate correctly over packet switched networks in which packets transmitted between the same two stations may traverse different routes (i.e., “datagram” networks).

3.2 CMTP Resource Optimization Strategy

A continuous media transport protocol should attempt to minimize the resources required by each continuous media stream while still meeting the real-time play-out deadlines at the receiver. Of the various resources required for guaranteed real-time delivery of a continuous media stream, the amount of buffer memory required at a set-top box is an important resource to be minimized. The importance of minimizing this buffer requirements is

receiver is due to the large number of set-top boxes which may potentially be in use in a large scale video-on-demand system.

As an example, consider a potential future scenario where set-top boxes are present in 40 million homes in America. Assume that memory costs \$30.00 per megabyte.² If the set-top box market is divided evenly among 10 manufacturers, then each manufacturer will have a total sales volume of 4 million set-top boxes. If the buffer memory required at the set-top box can be reduced from 1 megabyte to $\frac{1}{2}$ megabyte, the increase in profit to each set-top box manufacturer is 60 million dollars. When international sales are included in the analysis, the potential increase in profit from reducing the memory requirements in each set-top box is even greater. Therefore, a goal of the Continuous Media Transport Protocol will be to minimize the buffer requirements at the receiver.

The buffer requirements at a set-top box are minimized if the transmission schedule is such that the data is transmitted as late as possible while still guaranteeing that data is available at the receiver by its play-out deadline. As an example, consider the case of video transmission where each video frame is compressed to a constant size of 50 Kilobytes. If the server and transmission network are capable of transmitting data at a rate of 6 megabytes per second, then one second of video data (30 frames) could be transmitted in 0.25 seconds. In this case, a transmission schedule could be implemented where the data is transmitted in bursts which contain 1 second of video data, with the transmission time of a burst equal to 0.25

²A reasonable estimate in 1995 when purchased in large quantities.

seconds and an idle time between bursts of 0.75 seconds. However, this would require a buffer space of over 1 Megabyte at the receiver.³ A better alternative would be for the server to transmit each frame during the time that the receiver is displaying the previous frame. In this case, a set-top box would only require a buffer large enough to hold a single frame (i.e., 50 K bytes) before it is decompressed and displayed.

For most audio and video compression schemes, each frame does not, in general, compress to the same size. In such cases it may not always be possible to transmit a frame during the display time of the previous frame. (i.e., A large frame may require more transmission time than is available during the previous frame's display time.) For these cases, the buffer space at the receiver is still minimized when the data is transmitted as late as possible. A proof of this statement is given in Section 3.5.

Therefore, the Continuous Media Transport Protocol developed here uses a "*just in time*" transmission policy. That is, each frame is transmitted as late as possible while still guaranteeing that it is available at the receiver by its play-out deadline.

3.3 Data Structures Required at the Continuous Media Server

Although the Continuous Media Transport Protocol will be designed to minimize the

³This is calculated as follows:

Since there is 0.75 seconds of idle time between burst transmissions, at least

$$0.75 \text{ seconds} \times (30 \text{ frames/second}) \times (50 \text{ Kilobytes/frame}) = 1.125 \text{ Megabytes}$$

must be buffered. In fact, the buffer requirement is slightly greater than this since it will take some time to partially refill the receiver's buffer when a transmission burst starts.

buffer size required at the receiver, it is still necessary to verify that the receiver can allocate a buffer of sufficient size before transmission begins. Therefore, the protocol implements a resource negotiation process during the connection establishment phase. The resource negotiation process requires the continuous media server to be able to calculate bounds on the buffer space required at the receiver. Two data structures, the "*Delivery Deadline Table*" and the "*Transmission Schedule Table*", are used for this calculation. These data structures are explained in the following subsections.

3.3.1 The Delivery Deadline Table

In order to compute the resources required for a continuous media stream, the continuous media server must know the delivery deadlines for each frame. A simple table structure, termed a "*Delivery Deadline Table*", is used to represent this information. The entries in this table are indexed by frame number, and each entry contains fields which store the play-out time of the frame and the cumulative byte count of all frame sizes up to and including the current frame. The play-out times are considered to be offsets from the beginning of play-out. Thus, the first frame has a play-out time of zero. The size of an individual frame can be computed by subtracting the cumulative byte count field for the previous frame from the cumulative byte count field for the frame of interest.

The Delivery Deadline Table can be interpreted as a schedule of deadlines, where each entry in the schedule records the total amount of data which must have been received at a set-top box by the indicated deadline.

Table 3.1 illustrates the beginning of an example Delivery Deadline Table. In this

example, the frames are video frames with a play-back rate of 30 frames per second, and frames one through five have frame sizes of 30 K, 20 K, 25 K, 15 K, and 18 K bytes, respectively.

In general, the frame size values in a Delivery Deadline Table will vary, while the time interval between successive frames will be constant. However, for purposes of generality, the algorithms used by CMTP for calculation of resource requirements allow both to vary.

Table 3.1 Start of an Example Delivery Deadline Table

	Delivery Deadline (seconds)	Cumulative Number of Bytes
1	1/30	30 K
2	2/30	50 K
3	3/30	75 K
4	4/30	90 K
5	5/30	108 K

The Delivery Deadline Table for a continuous media file can be pre-computed and stored on disk at the server at the same time that the original continuous media file is loaded onto the server. Each entry in the Delivery Deadline Table requires several bytes (enough to represent a floating point number and an integer), while the compressed frame associated with an entry typically requires several Kilobytes. Thus, the storage overhead associated with the

Delivery Deadline Table is insignificant.

The server uses the Delivery Deadline Table to calculate the resource requirements for a continuous media stream; however, a graphical representation of this table will be used to illustrate the process. This graph is termed the “*Delivery Deadline Graph*”. A Delivery Deadline Graph plots the cumulative number of bytes required by the receiver in order to meet its play-out deadlines on the Y-axis, vs. time on the X-axis.

Figure 3.1 illustrates the Delivery Deadline Graph corresponding to the portion of the Delivery Deadline Table in Table 3.1. The Delivery Deadline Graph has a stair-step shape, corresponding to the fact that at discrete points in time the receiver removes a complete frame from its buffer, decompresses it, and plays it back. The height of each stair corresponds to the size of the frame. At any given time point on the X-axis, the Y-axis value of the Delivery Deadline Graph corresponds to the total number of bytes which must have been transmitted and placed in the receiver’s buffer in order to meet the play-out deadlines at the receiver.

3.3.2 The Transmission Schedule Table

The Delivery Deadline Table is pre-computed and stored on the server at the time its corresponding continuous media file is loaded. The server will use the Delivery Deadline Table to compute another table, termed the “*Transmission Schedule Table*”. Entries in the Transmission Schedule Table are indexed by frame number (similar to the Delivery Deadline Table). Each entry contains fields which store the time at which transmission of the frame must begin and the size of the frame. The transmission times are considered to be offsets from the beginning of transmission. Thus, the first frame has a transmission time of zero.

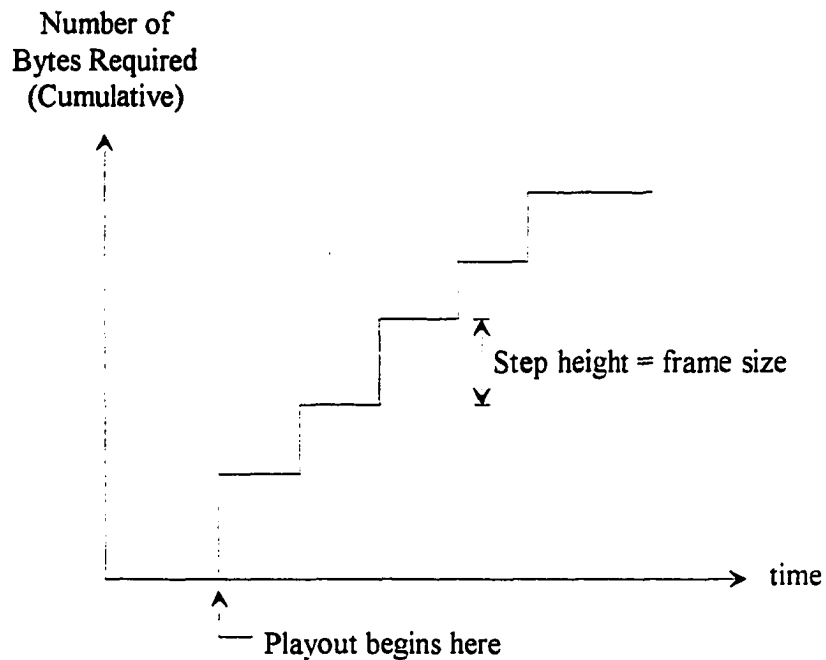


Figure 3.1. Delivery Deadline Graph

After computing the Transmission Schedule Table for a continuous media file, the server will use this table to schedule the transmission of each frame in the file. The Transmission Schedule Table is computed in such a way that the frames are transmitted as late as possible without violating their delivery deadlines at the receiver. As noted above (and proved in Section 3.5), this just-in-time transmission policy is used to minimize the buffer space requirements at the receiver.

A graphical representation of the Transmission Schedule Table can be used to illustrate the operation of the protocol. This graph is termed the "*Transmission Schedule Graph*". A Transmission Schedule Graph plots the cumulative number of bytes transmitted on the Y-axis

vs. time on the X-axis.

Figure 3.2 illustrates the start of an example Transmission Schedule Graph. Note that the Transmission Schedule Graph has a piece-wise linear shape. The segments of the graph which are flat represent time intervals during which no data is being transmitted. The segments of the graph which have a positive slope represent time intervals during which frames are being transmitted. The slope of these line segments corresponds to the network transmission rate.

Note also that the segments of the Transmission Schedule Graph with positive slope represent intervals during which a complete, integral number of frames are transmitted. The

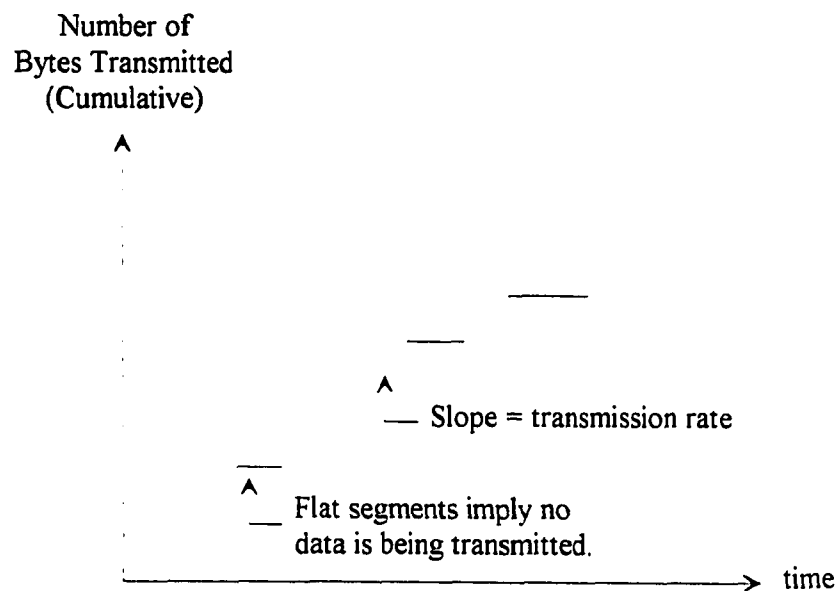


Figure 3.2. Transmission Schedule Graph

number of frames transmitted during an interval may be greater than one if the Transmission Schedule Table requires multiple frames to be sent back-to-back. However, once transmission of a frame has started, the complete frame is transmitted without interruption.

The axes of the Delivery Deadline Graph and the Transmission Schedule Graph are measured in the same units. Therefore, the graphs can be overlaid on top of each other, as shown in Figure 3.3. When this is done, the result can be interpreted as representing a producer-consumer relationship. The Transmission Schedule Graph represents the amount of data produced (i.e., the amount of data transmitted), while the Delivery Deadline Graph represents the amount of data consumed (i.e., the amount of data removed from the receiver's buffer, decompressed, and played out). In order to meet the real-time deadlines at the receiver, the value of the Transmission Schedule Graph should be greater than or equal to the value of the Delivery Deadline Graph at each point in time. (i.e., The amount of data transmitted should always be at least as much as that required by the receiver to meet its play-out deadlines.)

At any point in time the difference between the heights of the two graphs represents the amount of data in the receiver's buffer (i.e., the amount transmitted, but not yet consumed). Thus, the amount of buffer space required by the receiver can be determined by finding the maximum difference between the heights of the two graphs. A goal of the Continuous Media Transport Protocol is to compute a Transmission Schedule Table which minimizes the maximum difference between the heights of the two graphs.

The entries in the Delivery Deadline Table and the Transmission Schedule Table contain

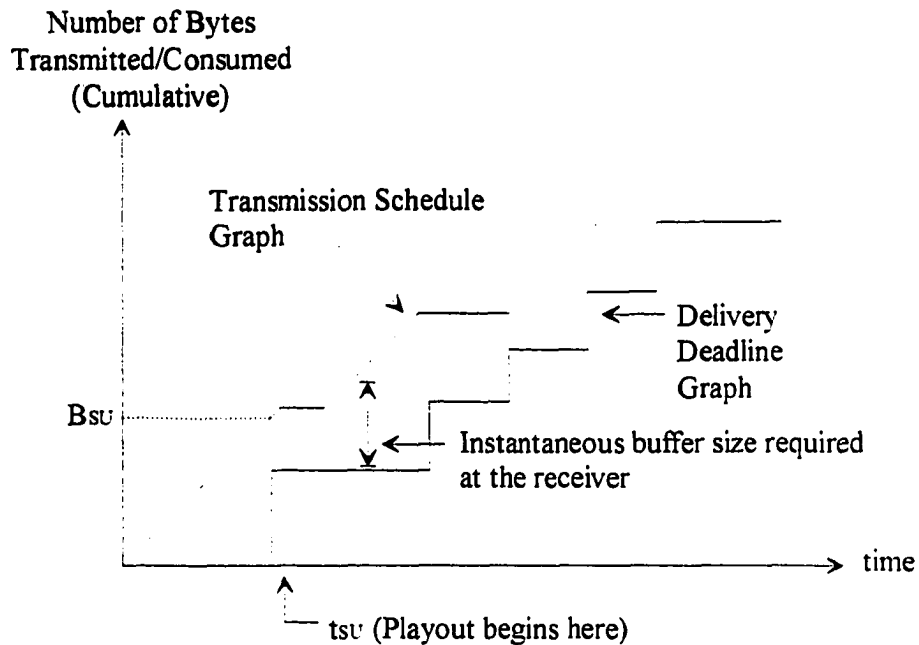


Figure 3.3. Overlaid Transmission Schedule Graph and Delivery Deadline Graph

time-stamp fields. The time-stamp fields in the Delivery Deadline Table record the play-out time for each frame. These times are offsets from the beginning of play-out. The time-stamp fields in the Transmission Schedule Table record the time at which transmission of each frame should start. These times are offsets from the beginning of transmission. A time of zero in the Delivery Deadline Table does not correspond to a time of zero in the Transmission Schedule Table since play-out cannot start before any data has been transmitted. The difference between the two time bases represents the start-up delay required to partially fill the receiver's buffer. This delay time is labeled as t_{su} in Figure 3.3. The value of the Transmission Schedule Graph at t_{su} corresponds to the amount of data which must be buffered at the receiver before play-

out can begin. This value is labeled B_{st} in Figure 3.3. During the connection establishment phase of the Continuous Media Transport Protocol, the value of B_{st} is transmitted to the set-top box. The set-top box will then monitor the filling of its buffer, and when B_{st} bytes have been received it will begin play-out.

We now state (a preliminary version of) the procedural steps that describe the operation of the Continuous Media Transport Protocol.

Protocol Steps for the Continuous Media Transport Protocol (Preliminary):

Given the Delivery Deadline Table for a continuous media file, the continuous media server should:

- A. Compute a Transmission Schedule Table which minimizes the buffer space required at the receiver.
- B. Calculate the (minimized) buffer size required at the receiver.
- C. Calculate the amount of data, B_{st} , which should be present in the receiver's buffer before play-out can begin.
- D. Transmit the information calculated in steps B and C to the receiver.
- E. If the receiver is able to allocate the required buffer space calculated in step B, the server should transmit the corresponding continuous media file using the transmission schedule calculated in step A. When B_{st} bytes have been buffered, the receiver will begin play-back of the media stream. If the receiver cannot allocate a buffer of the specified size, the server should refuse the connection request.

The reason that the steps listed above are labeled as “preliminary” is made clear in the next section where revised protocol steps are stated.

3.4 Calculation of Resource Requirements

As identified in Chapter 2, two types of network architectures will likely be used for wide-area continuous media distribution: those supporting a fixed transmission rate (e.g. T1) and those supporting a flexible transmission rate (e.g. ATM).

The method for calculating resource requirements differs for the two network architectures. For networks with a fixed transmission rate, the server knows the size and the delivery deadline of each frame from the Delivery Deadline Table. In addition, the server knows the network transmission rate. Using this information, it can calculate a Transmission Schedule Table which meets the real-time delivery deadlines for each frame, while still minimizing the buffer space required at the receiver. A procedure for calculating such a Transmission Schedule Table is given in Section 3.5.

For networks with a flexible transmission rate, however, a transmission schedule which minimizes the buffer requirements at the receiver will depend on the transmission bandwidth assigned to the media stream. For example, if the transmission bandwidth is large enough, then even the largest size frame can be transmitted during the play-out time of the previous frame. For this case, a buffer size equal to the size of the largest frame in the media stream is sufficient. However, if the transmission rate is too small, then it may not be possible to transmit every frame during the previous frame’s play-out period. As an example, consider a

trivial case of a continuous media file with only four frames. The sizes of frames one through four are 3 K, 1 K, 6 K, and 6 K bytes respectively, and each frame's play-out time is one second. If a transmission rate of 6 Kilobytes per second is chosen, then each frame can be transmitted during the play-out time of the previous frame, and thus a buffer size of 6 K bytes is sufficient at the receiver. However, if a transmission rate of 5 Kilobytes per second is used, the Transmission Schedule Graph in Figure 3.4 will minimize the buffer requirements at the receiver. (In this case, the data is transmitted as late as possible without violating the play-out time of the fourth frame).

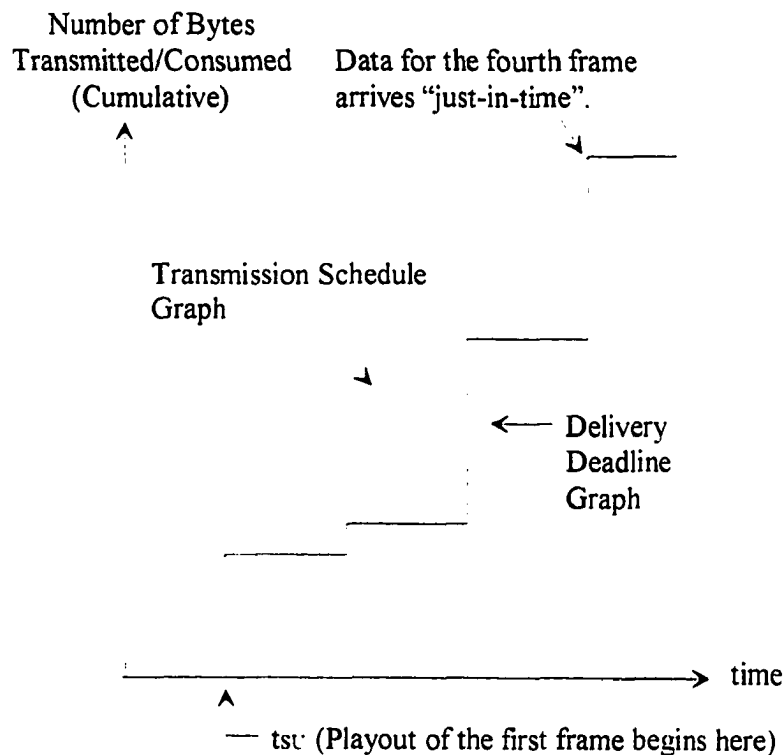


Figure 3.4. Transmission Schedule Graph for Data Rate of 5 Kilobits per second

Note that, in the example described above, just before the third frame is removed from the receiver's buffer, a buffer size of at least 7 K bytes is necessary. (6 K bytes are buffered for the third frame and 1 K bytes are buffered for the fourth frame; the remaining 5 K bytes of data for the fourth frame are transmitted during the play-out period of the third frame.)

Thus, if a transmission rate of 6 K bytes per second is selected for the Delivery Deadline Table in this example, then a buffer size of 6 K bytes is sufficient at the receiver. However, if a transmission rate of 5 K bytes per second is selected, a buffer size of 7 K bytes is required. So, for the case of a network with a flexible transmission rate, a Transmission Schedule Table which minimizes the buffer requirements at the receiver can not be calculated until a transmission rate has been selected. Therefore, when operating over a network which allows a transmission rate to be assigned at connection establishment time, the Continuous Media Transport Protocol follows a slightly different approach than when operating over a network with a fixed transmission rate. For networks with a flexible transmission rate, the receiver specifies its available buffer space during the connection request phase. The transmitter uses this information to compute the minimum transmission rate sufficient to meet the deadlines in the Delivery Deadline Table without overflowing the receiver's specified buffer size. The calculated transmission rate is then used to compute the Transmission Schedule Table. The details of this procedure are given in Sections 3.5 – 3.9.

Note that in the case of networks with a flexible transmission rate, the Continuous Media Transport Protocol will minimize the transmission rate assigned to the connection. This allows the server to maximize the number of connections supported across a given network

bandwidth.

We are now ready to state a revised list of steps that describe the operation of the Continuous Media Transport Protocol. There are two lists of steps depending on which network architecture (fixed transmission rate or flexible transmission rate) is to be used. The procedural steps for the case of a network with a fixed transmission rate are the same as those given earlier:

Protocol Steps for the Continuous Media Transport Protocol For Networks with Fixed Transmission Rates:

Given:

1. the Delivery Deadline Table for a continuous media file, and
2. the network transmission rate,

the continuous media server should:

- A. Compute a Transmission Schedule Table which minimizes the buffer space required at the receiver.
- B. Calculate the (minimized) buffer size required at the receiver.
- C. Calculate the amount of data, B_{SU} , which should be present in the receiver's buffer before play-out can begin.
- D. Transmit the information calculated in steps B and C to the receiver.
- E. If the receiver is able to allocate the required buffer space calculated in step B, the server should transmit the corresponding continuous media file using the Transmission Schedule Table calculated in step A. When B_{SU} bytes have been buffered, the receiver

will begin play-back of the media stream. If the receiver cannot allocate a buffer of the specified size, the server should refuse the connection request.

The new steps for the case of a network with a flexible transmission rate are as follows:

Protocol Steps for the Continuous Media Transport Protocol For Networks with Flexible Transmission Rates:

Given:

1. the Delivery Deadline Table for a continuous media file, and
2. the amount of buffer space available at the receiver,

the continuous media server should:

- A. Verify that enough buffer space is available at the receiver to hold the largest size frame.

If at least this much buffer space is available, it is possible to choose a transmission rate such that each frame can be transmitted during the play-out period of the previous frame. However, if the receiver's buffer space is smaller than the largest size frame, it is impossible to transmit the continuous media file for real-time play-out at the receiver. (i.e., The largest size frame cannot be buffered at the receiver no matter what transmission rate is selected.)

- B. If the receiver's buffer space is insufficient, refuse the connection request and abort the remaining steps.
- C. Calculate the minimum transmission rate required to transmit the continuous media file without violating the play-out deadlines in the Delivery Deadline Table.
- D. Calculate the Transmission Schedule Table corresponding to the transmission rate

calculated in step C.

- E. Calculate the amount of data, B_{SU} , which should be present in the receiver's buffer before play-out can begin.
- F. Attempt to establish a network connection with a bandwidth equal to that calculated in step C.
- G. If step F was successful, transmit the value of B_{SU} calculated in steps E to the receiver. If step F was unsuccessful, refuse the connection request and abort the remaining steps.
- H. Transmit the corresponding continuous media file using the Transmission Schedule Table calculated in step D. When B_{SU} bytes have been buffered, the receiver will begin play-back of the media stream.

The goal statements listed above require algorithms for the following four procedures:

1. Given a Delivery Deadline Table and a network transmission rate, calculate a Transmission Schedule Table which minimizes the buffer requirements at the receiver. (This is Step A for networks with a fixed transmission rate, and Step D for networks with a flexible transmission rate.) An algorithm which accomplishes this function is given in Section 3.5, along with a proof that the resulting Transmission Schedule Table minimizes the buffer space required at the receiver.
2. Given a Delivery Deadline Table and a Transmission Schedule Table, calculate the buffer space required at the receiver. (This is Step B for networks with a fixed transmission rate.) An algorithm which accomplishes this function is given in Section 3.7.

3. Given a Delivery Deadline Table and a Transmission Schedule Table, calculate the amount of data, B_{st} , which must be present in the receiver's buffer before play-out can begin. (This is Step C for networks with a fixed transmission rate, and Step E for networks with a flexible transmission rate.) An algorithm which accomplishes this function is given in Section 3.8.
4. Given a Delivery Deadline Table and the amount of buffer space available at the receiver, calculate the minimum transmission rate sufficient to transmit the continuous media file without violating the play-out deadlines in the Delivery Deadline Table. (This is Step C for networks with a flexible transmission rate.) An algorithm which accomplishes this function is given in Section 3.9.

3.5 Calculation of the Transmission Schedule Table

In this section, a procedure for calculating a Transmission Schedule Table which minimizes the buffer requirements at the receiver is given. An algorithm which performs this procedure is presented using the C programming language. In addition, the steps of this algorithm are illustrated by the construction of a Transmission Schedule Graph which corresponds to the Transmission Schedule Table computed by the algorithm. Finally, a simple proof that the stated algorithm actually minimizes the buffer requirements at the receiver is presented.

The algorithm presented in this section requires the server to have access to the Delivery Deadline Table for the continuous media file to be transmitted, and the network

transmission bandwidth to be used for transmission of the file. If the network allows the transmission bandwidth to be specified during the connection establishment phase, then the procedure in Section 3.9 must first be used to compute the minimum bandwidth sufficient to transmit the file.

In order to illustrate the algorithm, the construction of the Transmission Schedule Graph is described first. This graph is constructed segment by segment, with the segments superimposed on the corresponding Delivery Deadline Graph as they are constructed. The construction of the Transmission Schedule Graph starts with the last segment and works backwards. As each new segment is constructed, it is overlaid on the Delivery Deadline Graph. When the procedure has finished, a complete Transmission Schedule Graph will be superimposed upon the Delivery Deadline Graph.

In order to minimize the buffer requirements at the receiver, the Transmission Schedule Graph which is constructed implements a “just in time” delivery policy. The procedure starts by computing the point on the Transmission Schedule Graph at which data transmission must begin in order to meet the delivery deadline for the last frame. Graphically, this is represented by extending a line segment from the delivery deadline for the last frame of the Delivery Deadline Graph backwards in time. The slope of the line segment corresponds to the network transmission rate. Since the Transmission Schedule Graph can never fall below the Delivery Deadline Graph, the line segment must stop if it reaches a point where it intersects the Delivery Deadline Graph.

Figure 3.5 illustrates the procedure. In this (trivial) example, there are five frames. The

first segment (working backwards in time) of the Transmission Schedule Graph has been constructed to meet the play-out deadline for frame 5. This segment also happens to meet the play-out deadline for frame 4. This can be seen by the fact that the line segment lies above the Delivery Deadline Graph at the point in time when play-out of frame 4 begins. In fact, when frame 4 is removed from the receiver's buffer and played out, there is already some data buffered for frame 5. However, this cannot be avoided without missing the play-out deadline for frame 5.

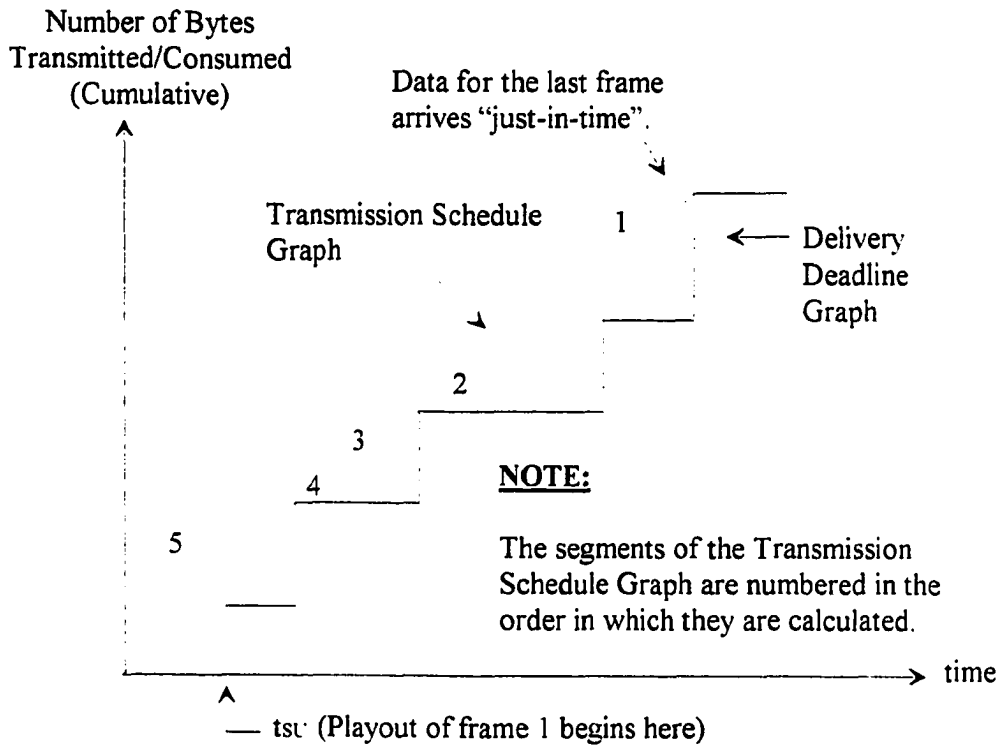


Figure 3.5. Calculation of a Transmission Schedule Graph

Note that if this first line segment were extended backwards any further, the deadline for frame 3 would be missed. This can be seen by the fact that such a line segment would fall under the Delivery Deadline Graph at the time in which frame 3 is played out. In order to meet the delivery deadline for frame 3, data transmission must start at an earlier time than is required to meet the delivery deadlines for frames 4 and 5. Thus, in order to meet the delivery deadline for frame 3, the next segment of the Transmission Schedule Graph (working backwards) must be a flat segment. Such a segment corresponds to a time interval when no data is to be transmitted. This flat segment extends backwards in time from the end point of the initial segment of the Transmission Schedule Graph to the point in time when play-out of frame 3 starts.

The next segment of the Transmission Schedule Graph (again, working backwards) is calculated to meet the “just-in-time” deadline for the play-out of frame 3. Thus, this segment has a positive slope, and extends backwards in time from the point on the Delivery Deadline Graph where play-out of frame 3 begins to the point where this next segment intersects the Delivery Deadline Graph. As can be seen from Figure 3.5, if this segment were extended backwards any further, the delivery deadline for frame 2 would be missed.

The final two segments are constructed in a similar manner. As can be seen from Figure 3.5, the segment which meets the delivery deadline for frame 2 also happens to meet the delivery deadline for frame 1.

In the final Transmission Schedule Graph of Figure 3.5, the segment numbered “5” corresponds to the interval in time when frames 1 and 2 are to be transmitted. These frames

will be transmitted back-to-back, with frame 2 arriving just in time to meet its play-out deadline. Frame 1 will arrive earlier than required to meet its play-out deadline, however this is necessary in order for the play-out deadline of frame 2 to be met. Similarly, the segment numbered “3” represents the interval during which frame 3 is to be transmitted in order to meet its play-out deadline, and the segment numbered “1” represents the interval in time during which frames 4 and 5 are to be transmitted. These frames are transmitted back-to-back, and are to be transmitted such that frame 5 arrives just in time to meet its play-out deadline. In order to meet this deadline, it is necessary for frame 4 to arrive early. The segments numbered “2” and “4” represent intervals during which no data is to be transmitted.

A formal description of the procedure illustrated above is presented next. The notation used is as follows: Assume there are n frames, numbered 1 through n . Let $s[i]$ be the size of frame i , and let $d[i]$ be its play-out deadline. Note that $d[i]$ is stored in the Delivery Deadline Table and that $s[i]$ can be calculated from the Delivery Deadline Table by subtracting the cumulative byte count for frame $i - 1$ from the cumulative byte count for frame i . Let B be the network transmission bandwidth. For $i = 1, \dots, n$ the algorithm must compute the time, $t[i]$, at which transmission of frame i must begin. These computed values will be used to form the Transmission Schedule Table.

The values of $t[i]$ are calculated in two passes. During the first pass, a time value, $p[i]$, is assigned to each frame. The value of $p[i]$ represents the amount of time, measured prior to play-out of the last frame, at which transmission of frame i should start. Note that as the value of i increases, the value of $p[i]$ decreases. For example if $p[49] = 105.2$ and $p[50] = 105.05$,

then transmission of frame 49 should begin 105.2 seconds before the play-out of the last frame, and transmission of frame 50 should begin 105.05 seconds before the play-out of the last frame. During the second pass, the values of $t[i]$ are calculated using the values of $p[i]$ computed during the first pass.

A summary of this notation is given in Table 3.2. A C language algorithm to compute the Transmission Schedule Table is presented in Figure 3.6. A simple proof that the just-in-time delivery policy used by the algorithm in Figure 3.6 to construct the Transmission Schedule Table minimizes the buffer requirements at the receiver is given in Theorem 3.1 below.

Table 3.2 Notation for Calculation of the Delivery Deadline Table

Variable	Meaning
B	Network transmission bandwidth
n	Total number of frames
i	Index for current frame
$s[i]$	Size of frame i (from Delivery Deadline Table)
$d[i]$	Play-out deadline of frame i (from Delivery Deadline Table)
$p[i]$	Time prior to play-out of frame n , at which transmission of frame i should begin
$t[i]$	Time at which transmission of frame i should begin

```

/*
   t represents the amount of time between the end of
   transmission for the current frame and the beginning
   of play-out of frame n (i.e., the last frame).
*/
t = 0;      /* last frame can begin play-out as soon as it
             is received. */

/* First pass, calculate the values of p[i]. */
for (i = n; i > 0; i--)
{
    /*
       Calculate p[i] by adding the transmission time for
       frame i to the amount of time, t, between the end
       of transmission for frame i and the beginning of
       play-out of frame n.
    */

    p[i] = (s[i] / B) + t;

    /*
       Do we need to add a "flat segment" to the
       the Transmission Schedule Graph?
    */

    if (p[i] < (d[n] - d[i - 1])) /* Yes */
        t = d[n] - d[i - 1];
    else /* No */
        t = p[i];
}

/* Second pass, calculate the values of t[i]. */
for (i = 1; i <= n; i++)
{
    t[i] = p[1] - p[i];
}

```

Figure 3.6 C Language Algorithm for Constructing the Transmission Schedule Table

Theorem 3.1:

If the Transmission Schedule Table constructed by the algorithm presented in Figure 3.6 is used by the Continuous Media Transport Protocol to transmit a continuous media file, then the buffer space required at the receiver is minimized.

Proof:

At any point in time, t , let $P(t)$ be the total amount of data which has been transmitted by the server and placed in the receiver's buffer, and $C(t)$ be the total amount of data which has been removed from the buffer for consumption by the receiver. Note that for a given continuous media file, the value of $C(t)$ is determined solely by the point in time at which play-out of the file begins at the receiver.

At time t , the amount of data in the receiver's buffer is equal to $P(t) - C(t)$. This value is minimized when $P(t)$ is minimized and $C(t)$ is maximized. However, $P(t)$ is minimized if data is sent as late as possible without violating a play-out deadline, and $C(t)$ is maximized if play-out begins as soon as a sufficient amount of data has been buffered by the receiver. Both conditions hold for the Continuous Media Transport Protocol when data transmission is scheduled using the just in time Transmission Schedule Table computed by the algorithm presented in Figure 3.6.

3.6 Calculation of t_{st}

Recall from Section 3.3.2 that the time bases for the Delivery Deadline Table and the

Transmission Schedule Table differ. The time-stamp values in the Transmission Schedule Table represent the times at which transmission of each frame should start, and therefore are offsets measured from the beginning of transmission. The time-stamp values in the Delivery Deadline Table represent the times at which play-out of each frame should begin, and therefore are offsets measured from the beginning of play-out. In order to calculate the buffer requirements at the receiver, it is necessary to know the difference between these two time bases. This difference is labeled as t_{st} in Figure 3.3.

The difference in the two time bases can be calculated by comparing a time value from the Transmission Schedule Table to a time value from the Delivery Deadline Table at a point where both values are known. The play-out deadline of the last frame can be used as such a reference point. This time value, measured from the Delivery Deadline Table's time-base, is directly available from the Delivery Deadline Table. The corresponding value, measured from the Transmission Schedule Table's time-base, can be calculated by adding the scheduled transmission time of the last frame (available from the Transmission Schedule Table) to the transmission duration of the last frame. The transmission duration of the last frame can be determined by dividing the size of the last frame by the network transmission rate:

$$t_{st} = t[n] + \left(\frac{s[n]}{B} \right) - d[n] \quad (3.1)$$

Note that the value of $t[n] + \left(\frac{s[n]}{B} \right)$ in Equation 3.1 represents the arrival time of the last frame as measured by the transmitter's time base. Since the last frame is scheduled to be

transmitted such that it arrives at precisely its play-out deadline, this value can be compared to the receiver's time base as measured by the play-out deadline of the last frame in order to compute the value t_{su} . A frame which is scheduled to arrive earlier than its play-out deadline could not be used for calculating t_{su} in Equation 3.1, since, for such a frame, the arrival time as measured by the transmitter's time base does not correspond to the play-out time as measured by the receiver's time base.

The value of t_{su} which was calculated in Equation 3.1 can be added to the time-stamp values in the Delivery Deadline Table in order that these time-stamps are referenced to the same time base as the time-stamps in the Transmission Schedule Table. Such a translation will be required by the algorithms presented in Sections 3.7 and 3.8.

3.7 Calculation of the Buffer Space Required at the Receiver

In this section, a procedure for calculating the buffer space required at the receiver is presented.

At each point in time, the amount of buffer space required at the receiver can be determined by subtracting the value of the Delivery Deadline Graph from the value of the Transmission Schedule Graph. In general, this difference will vary with time. In order to guarantee that the buffer space at the receiver is sufficient for transmission and play-out of the complete continuous media file, it is necessary to find the maximum difference between the heights of the two graphs.

The maximum difference between the height of the Transmission Schedule Graph and

the Delivery Deadline Graph must occur at an instant just prior to the removal of a frame from the receiver's buffer. This is due to the fact that during the play-out interval of a single frame the value of the Transmission Schedule Graph is non-decreasing while the value of the Delivery Deadline Graph remains constant. Thus, during the play-out interval for a frame, the difference between the heights of the two graphs has its maximum value at the end of the interval. Therefore, in order to compute the maximum difference between the heights of the two graphs across the play-out duration of the complete continuous media file, it is sufficient to compute the difference at each point in time just prior to play-out of a frame, and then find the maximum of these values.

At the instant just prior to play-out of frame i , the difference between the heights of the Transmission Schedule Graph and Delivery Deadline Graph corresponds to the amount of data which has been transmitted beyond the last byte of frame $i - 1$. Stated another way, just prior to play-out of frame i , the difference in the heights of the Transmission Schedule Graph and the Delivery Deadline Graph corresponds to the amount of data which has been transmitted starting with the first byte of frame i . In order to compute this value, the following Theorem will be used:

Theorem 3.2:

If the algorithm presented in Figure 3.6 is used to compute the Transmission Schedule Table for a continuous media file, then the following condition holds: For every frame, i , during the time interval between the start of transmission of frame i and the instant in time that frame i is removed from the receiver's buffer for play-out, data is transmitted without

interruption. (i.e., There are no flat segments on the Transmission Schedule Graph during the interval between the start of transmission of a frame and the delivery deadline for that frame.)

Proof:

By the algorithm used to construct the Transmission Schedule Graph, a flat segment can only be positioned in time between the delivery deadline for some frame, j , and the start of transmission of frame $j + 1$. Thus, for frame i , during the time interval corresponding to such a flat segment, the delivery deadline for frame i has either already occurred (if $i \leq j$), or transmission of frame i has not yet started (if $i \geq j + 1$). Therefore, if transmission of frame i has already started (that is, if $i \leq j$), then its delivery deadline has already occurred. \square

Thus, just before play-out of frame i , the amount of data in the receiver's buffer can be determined by multiplying the amount of time between the start of transmission of frame i and the play-out deadline of frame i by the network transmission rate. The amount of time between the start of transmission of frame i and its play-out deadline can be computed by subtracting $t[i]$, the time at which frame i is scheduled to be transmitted, from $(d[i] + t_{st})$, its delivery deadline referenced to the same time base as $t[i]$. Thus, the buffer space required at the receiver can be calculated as follows:

$$\text{Buffer space} = \max\{(d[i] + t_{st} - t[i]) \times B\} \quad i = 1, \dots, n \quad (3.2)$$

3.8 Calculation of B_{st}

During the connection establishment phase, the continuous media server notifies the

client of the amount of data which should be in its receive buffer before play-out begins. This value is labeled as B_{SU} in Figure 3.3.

A method that the server can use to calculate B_{SU} follows directly from the derivation of the receiver's buffer space requirements described in Section 3.7. That is, B_{SU} can be interpreted as the difference between the height of the Transmission Schedule Graph and the Delivery Deadline Graph at the instant that the first frame is to be removed from the receiver's buffer. Therefore:

$$B_{SU} = (d[1] + t_{SU} - t[1]) \times B = t_{SU} \times B \quad (3.3)$$

The simplification in Equation 3.3 results from the fact that $d[1]$ and $t[1]$ are both 0.

3.9 Calculation of the Minimum Transmission Rate for a Fixed Size Receive Buffer

For network architectures which support a flexible transmission rate, the operation of the Continuous Media Transport Protocol requires that a set-top box transmit its available buffer size to the continuous media server during the connection establishment phase. The continuous media server will use this buffer size to calculate the minimum transmission rate sufficient to guarantee that a Transmission Schedule Table can be constructed which meets the play-out deadlines of the requested continuous media file. This section presents an algorithm which can be used to calculate this transmission rate.

A graph, termed the "*Transmission Limit Graph*" will be used to illustrate the algorithm. The Transmission Limit Graph plots the maximum amount of data which may be

transmitted without overflowing the receiver's specified buffer size on the Y-axis vs. time on the X-axis.

Recall that at any point in time the height of the Delivery Deadline Graph represents the amount of data which has been removed from the receiver's buffer. If S represents the size of this buffer, then at each point in time the height of the Transmission Limit Graph is equal to the height of the Delivery Deadline Graph plus S . That is, at any point in time the amount of data which may be transmitted without overflowing the receiver's buffer is equal to the amount of data which will have been removed from the buffer by that time (i.e., the value of the Delivery Deadline Graph) plus the size of the receiver's buffer (i.e., value of S).

Given a Delivery Deadline Graph and its corresponding Transmission Limit Graph, we define a *feasible* Transmission Schedule Graph to be one which, at each point in time, has a value greater than or equal to the value of the Delivery Deadline Graph and less than or equal to the value of the Transmission Limit Graph (Figure 3.6). Such a Transmission Schedule Graph represents a transmission schedule which meets the play-out deadlines for each frame without overflowing the allotted buffer space at the receiver. Similarly, we define a *feasible* transmission rate as a transmission rate from which it is possible to construct a feasible Transmission Schedule Graph. The algorithm presented in this section calculates the minimum feasible transmission rate, that is the minimum transmission rate for which it is possible to construct a Transmission Schedule Graph which meets the play-out deadlines at the receiver without overflowing the specified buffer size at the receiver.

The following notation will be used: Let $DDG(t)$ represent the height of the Delivery

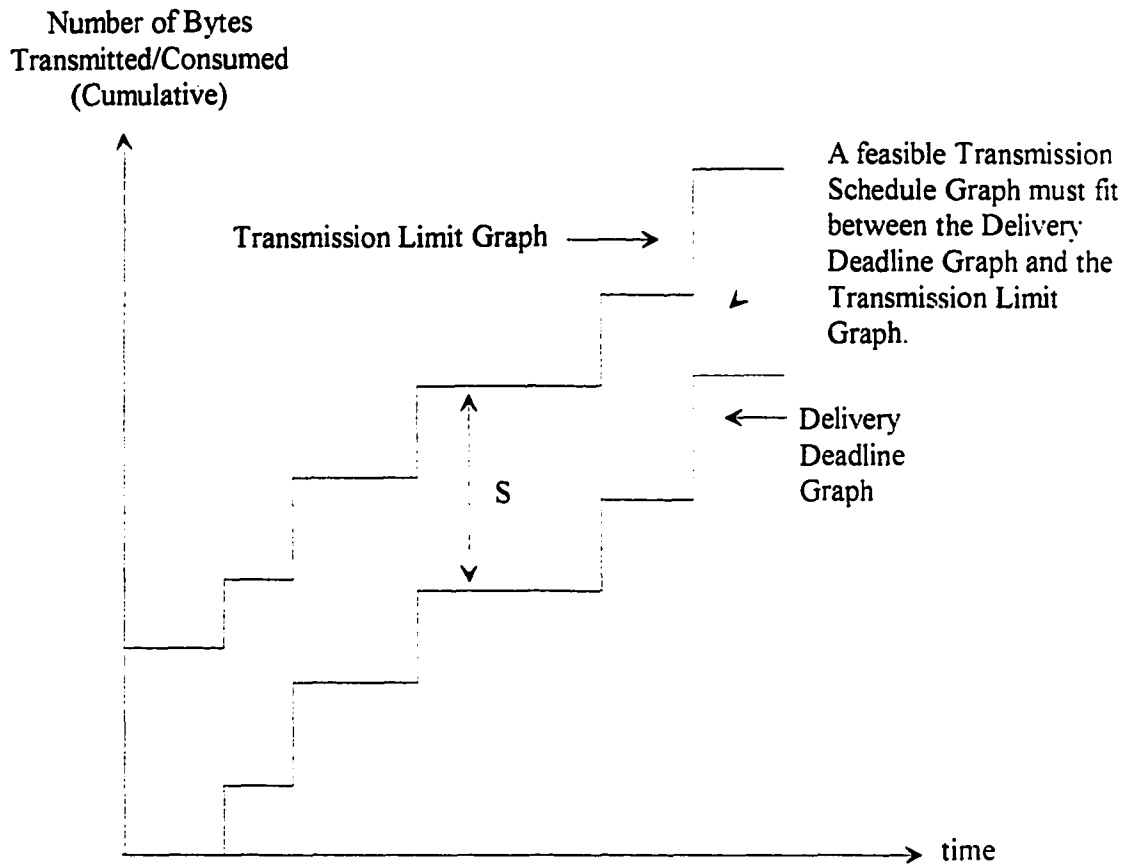


Figure 3.7. An Example Transmission Limit Graph

Deadline Graph at time t . Technically, this value is not well defined at points in time where play-back of a new frame begins, since at these points there is a discontinuity in the Delivery Deadline Graph. Therefore, the value of $DDG(t)$ at such points will be taken as the value at $DDG(t^-)$, corresponding to the height of the Delivery Deadline Graph immediately after a step discontinuity. $DDG(t)$ can be interpreted as the amount of data which must be present in the receiver's buffer at time t . The definition of $DDG(t)$ as $DDG(t^-)$ at the points where the

graph is discontinuous is consistent with the fact that data for a frame must be present in the receiver's buffer by the instant that play-out of that frame is to begin.

Similarly let $TLG(t)$ represent the height of the Transmission Limit Graph at time t . At the points in time where play-back of a new frame begins, however, the value of $TLG(t)$ will be defined to be the value of $TLG(t^-)$, corresponding to the height of the Transmission Limit Graph immediately before the step discontinuity. $TLG(t)$ can be interpreted as the maximum amount of data which may be transmitted and placed in the receiver's buffer by time t . This definition of $TLG(t)$ as $TLG(t^-)$ at points where the graph is discontinuous is consistent with the fact that a practical decoder cannot instantaneously remove a frame from its buffer at the instant that that play-back of a frame is to begin.

Thus, for any point in time except those points corresponding to the time that play-back of a new frame begins, $TLG(t) = DDG(t) + S$. However, at the points where the Delivery Deadline Graph (and thus the Transmission Limit Graph) are discontinuous, $TLG(t) \neq DDG(t) + S$.

Finally, let $TSG(t)$ represent the value of the Transmission Schedule Graph at time t . This graph is always continuous, and therefore there is no need to further define $TSG(t)$ at points in time where play-back of a new frame begins.

Given a Delivery Deadline Graph and a Transmission Limit Graph (or equivalently, a Delivery Deadline Graph and a value of S) we must find the minimum transmission rate for which a Transmission Schedule Graph can be drawn such that the value of the Transmission

Schedule Graph is greater than or equal to $DDG(t)$ and less than or equal to $TLG(t)$ for each value of t .

The following theorem places a lower bound on the set of feasible transmission rates:

Theorem 3.3:

Given any two points in time, t_1 and t_2 with $t_1 < t_2$, a feasible transmission rate, r , must satisfy the following inequality:

$$r \geq \left(\frac{DDG(t_2) - TLG(t_1)}{t_2 - t_1} \right) \quad (3.4)$$

Proof:

The following sequence of inequalities will be used in the proof:

For a feasible Transmission Schedule Graph, the value of the Transmission Schedule Graph must always be greater than or equal to the value of the Delivery Deadline Graph. Therefore, it is required that:

$$DDG(t_2) \leq TSG(t_2) \quad (3.5)$$

The value of $TSG(t_1)$ represents the amount of data which has been transmitted and placed in the receiver's buffer at time t_1 . Similarly, the value of $TSG(t_2)$ represents the amount of data which has been transmitted and placed in the receiver's buffer at time t_2 . For a transmission rate of r , the following inequality must hold:

$$TSG(t_2) \leq TSG(t_1) + (r \times (t_2 - t_1)) \quad (3.6)$$

Equation 3.6 will be an equality if data is transmitted during the complete interval from t_1 to t_2 . However, since a Transmission Schedule Graph may have one or more flat segments in this time interval, the equation is written an inequality.

Finally, since the value of the Transmission Schedule Graph must always be less than or equal to the value of the Transmission Limit Graph at every point:

$$TSG(t_1) + (r \times (t_2 - t_1)) \leq TLG(t_1) + (r \times (t_2 - t_1)) \quad (3.7)$$

Applying the transitive property on the inequality sequence in Equations 3.5 – 3.7 leads to:

$$DDG(t_2) \leq TLG(t_1) + (r \times (t_2 - t_1)) \quad (3.8)$$

Finally, solving the inequality in Equation 3.8 for r yields:

$$r \geq \left(\frac{DDG(t_2) - TLG(t_1)}{t_2 - t_1} \right) \quad \dots$$

Theorem 3.3 can be interpreted as follows: For a given Delivery Deadline Graph and Transmission Limit Graph, a feasible transmission rate cannot be smaller than the slope of any line segment between a point on the Transmission Limit Graph and a point further out in time on the Delivery Deadline Graph. Thus, it is necessary for a feasible transmission rate, r , to be

at least as large as the maximum of the slopes of all such line segments between the two graphs.

It will be proven in Theorem 3.5 that a transmission rate which is as large as this maximum slope is also sufficient for constructing a feasible Transmission Schedule. That is, such a transmission rate is a *feasible* transmission rate. Since such a transmission rate is feasible, but by Theorem 3.3, no smaller transmission rate is feasible, this rate corresponds to the minimum feasible transmission rate. Thus, it is this rate which should be used to construct the Transmission Schedule Table.

In order to compute this minimum feasible transmission rate, it is necessary to find the line segment from a point in time on the Transmission Limit Graph to a point further out in time on the Delivery Deadline Graph that has the largest slope of all such line segments. The following theorem can be used to find this line segment:

Theorem 3.4 :

The line segment with the largest slope from time t_1 on the Transmission Limit Graph to a time $t_2 > t_1$ on the Delivery Deadline Graph occurs when the end points, t_1 and t_2 correspond to the play-out deadlines of two frames.

Proof:

Inside a play-out interval, the heights of the Transmission Limit Graph and the Delivery Deadline Graph are constant. That is, for any value of t_1 within a play-out interval on the Transmission Limit Graph, $TLG(t_1)$ is constant. Similarly, for any value of t_2 inside a play-out interval on the Delivery Deadline Graph, $DDG(t_2)$ is constant. The line segment between

these points has slope:

$$r = \left(\frac{DDG(t_2) - TLG(t_1)}{t_2 - t_1} \right).$$

This value will have its largest value when t_2 is as small as it can be, corresponding to the start of its play-out interval on the Delivery Deadline Graph and t_1 is as large as it can be, corresponding to the end of its play-out interval on the Transmission Limit Graph.

Thus (based on Theorem 3.5 below), the minimum transmission rate can be found by computing the slope of all line segments between the end of a play-out interval on the Transmission Limit Graph and the start of a subsequent play-out interval on the Delivery Deadline Graph and taking the maximum of these slopes. The X and Y coordinates of each point at the start of a play-out interval on the Delivery Deadline Graph are stored in the Delivery Deadline Table. The X-coordinate of a point at the end of a play-out interval on the Transmission Limit Graph is identical to the X-coordinate of the start of the following play-out interval, and therefore is also available from the Delivery Deadline Table. The Y-coordinate of a point at the end of a play-out interval on the Transmission Limit Graph can be found by adding the value of the receiver's buffer size, S , to the cumulative byte count field for that frame, and is thus easily calculated from the Delivery Deadline Table. A C language algorithm to compute the minimum feasible transmission rate is presented in Figure 3.8.

Theorem 3.5, below, proves that the transmission rate constructed by the algorithm in Figure 3.8 is a feasible transmission rate. As noted earlier, no transmission rate less than the

```
/*
   n represents the number of frames. DDG[x] and TLG[x]
   are computed from the Delivery Deadline Table and the
   receiver's buffer size, where DDG[x] is the cumulative
   byte count for frame x from the Delivery Deadline
   Table, and TLG[x] is equal to DDG[x] + S, the
   receiver's buffer size.
*/

max_slope = 0;

for (I = 1; I <= n; I++)
    for (j = I+1; j <= n; j++)
    {
        slope = (DDG[j] - TLG[i]) / (d[j] - d[i]);
        if (slope > max_slope)
            max_slope = slope;
    }

r = max_slope;
```

Figure 3.8 C Language Algorithm for Computing the Minimum Feasible Transmission Rate

value computed by this algorithm is feasible. Thus, the computed rate is actually the minimum feasible transmission rate.

Theorem 3.5 :

The transmission rate computed by the algorithm in Figure 3.8 is a feasible transmission rate.

Proof:

The algorithm computes the maximum slope, r , of all line segments between a point on the Transmission Limit Graph to a point further out in time on the Delivery Deadline Graph. If this slope, r , is used to construct a Transmission Schedule Graph, all non-flat line segments of this graph will have a slope r and will be constructed by extending the line segment from a play-out deadline on the Delivery Deadline Graph, backwards in time. The Transmission Schedule Graph so constructed will be a feasible Transmission Schedule Graph if no such line segment lies above the Transmission Limit Graph at any point in time.

Assume that a Transmission Schedule Graph was constructed using the value of r for the slope of the non-flat line segments and that a line segment of this graph did lie above the Transmission Limit Graph at some point in time, t_x . Then the slope of such a line segment would not correspond to the maximum slope of all line segments between a point on the Transmission Limit Graph and a point further out in time on the Delivery Deadline Graph, since a line segment between the original point on the Delivery Deadline Graph and the point on the Transmission Limit Graph corresponding to time t_x would have a slope greater than r . This contradicts the assumption that r is the maximum slope of all such line segments. Thus,

no line segment constructed with slope r will lie above the Transmission Limit Graph. and thus, r is a feasible transmission rate.

CHAPTER 4. CMTP REFINEMENTS FOR NON-IDEAL NETWORKS

In this chapter, the Continuous Media Transport Protocol is refined in order that the ideal assumptions listed in Section 3.1 can be removed. The refinements will allow the protocol to operate in realistic networking environments.

4.1 Protocol Refinements to Compensate for Network Jitter

Jitter is defined as the difference between the maximum and the minimum packet transmission delays. A non-zero value of jitter may result in the data for a frame arriving after its play-out deadline. For example, consider that a receiver begins play-out of the first frame in a continuous media stream when its buffer has filled to a certain value, B_{sc} . Assume that the data for the first B_{sc} bytes experiences the smallest possible network transmission delay, that the last byte of the second frame is not contained in the first B_{sc} bytes, and that packet containing this byte experiences the largest possible network delay. If the second frame is scheduled to arrive “just-in-time” under the assumption of a constant network delay, then it would arrive beyond its scheduled play-back time in the scenario described above. In fact, it would have missed its play-out deadline by an amount of time equal to the difference between the maximum and minimum possible network transmission delays.

To compensate for network jitter, therefore, it is necessary for the receiver to delay the play-back of the first frame beyond the point at which playback would begin under the ideal environment assumed in Chapter 3. In the worst case, the receiver must delay play-back under

the assumption that the first B_{SC} bytes experienced the minimum possible network transmission delay, while subsequent bytes may experience the maximum possible network transmission delay. Therefore, to compensate for network jitter, the refined Continuous Media Transport Protocol requires the receiver to delay starting play-back by an amount of time equal to the maximum network jitter after the first B_{SC} bytes have been received. Furthermore, additional buffer space is required at the receiver to serve as a *jitter-smoothing* buffer. If the network jitter is bounded by J_{MAX} , and the transmission rate is r , this buffer space is bounded by $r \times J_{MAX}$. The rules which must be added to the protocol to compensate for jitter are as follows:

1. A receiver should delay play-back until an amount of time J_{MAX} , after the first B_{SC} bytes have been received.
2. A receiver should allocate an additional $r \times J_{MAX}$ bytes of buffer space beyond the buffer size required for the ideal network environment. This buffer space will serve as a jitter-smoothing buffer.

4.2 Protocol Refinements to Compensate for Clock Frequency Errors

This section extends the Continuous Media Transport Protocol to compensate for differences between the clock frequencies of the local oscillators at the continuous media server and a set-top box.

As discussed in Section 2.2.1, it is possible for the errors in the clock frequencies at the server and a set-top box to lead to a gradual overrun or underrun of data at the set-top box's

receive buffer if a mechanism to compensate for the clock frequency errors is not designed into the protocol. If the server knew the exact frequency of the receiver's clock, it could compensate for any errors in this clock frequency by calculating a Transmission Schedule Table based on the receiver's actual clock frequency.⁴ However, an oscillator frequency is typically known only to within a certain tolerance. Therefore, it is not possible for the server to know the exact clock frequency at the receiver. Instead, the protocol must correct for slight errors in this clock frequency by observing the difference between the actual and expected play-out rates at the receiver and performing corrective action as necessary.

In the method proposed here, the receiver will send occasional feedback messages to the transmitter to report the difference between the expected and actual fullness values of its buffer. The server will use this information to make slight adjustments to its transmission schedule.

Since the server does not know the actual play-back rate at the receiver, it must construct its original transmission schedule based on the assumption that the receiver's clock is running at its fastest possible rate. For example, if the nominal play-out rate is 30 frames per second, but the clock tolerance at the receiver is 200 parts per million (relative to the transmitter's clock), then the transmitter must assume that the play-out interval of each frame is $\left(\frac{1}{30} \times 0.9998\right)$ seconds. This will guarantee that the receiver does not experience an

⁴ In the calculations performed in this Section, the transmitter's clock is considered as a reference clock with no frequency errors. Thus, all errors due to non-zero clock frequency tolerances are considered to be lumped as a tolerance error of the receiver's clock relative to the passage of time as measured by the transmitter's clock.

underrun error during play-back; however, it may lead to a gradual accumulation of data in the receiver's buffer if the receiver's play-back clock is running at a somewhat slower rate. In order to correct for this gradual accumulation of data, the continuous media server will use the information in the feedback messages received from a set-top box to insert extra idle time into the transmission schedule when necessary.

This mechanism requires that a set-top box be able to determine the expected value of the fullness of its buffer at certain points in time. Given a Transmission Schedule Table and a Delivery Deadline Table, the algorithm presented in Section 3.7 was used by the server to calculate the expected buffer fullness at the points in time corresponding to the delivery deadline for each frame. In the protocol extension described here, the expected buffer fullness values computed by this algorithm will be used to provide the checkpoint values for comparisons to the actual buffer fullness at the receiver. The details of this procedure are described below.

The value of the expected buffer fullness as computed by the server will be sent in a fixed-length header that is appended to the start of each frame.⁵ When the start of a play-back interval for a frame occurs at the receiver, the receiver will check its actual buffer fullness against the expected buffer fullness value in the frame's header. If the difference is greater than a threshold value, the receiver will send a feedback message to the server to indicate the

⁵ The length of this header must be accounted for in the Delivery Deadline Table and Transmission Schedule Table. Thus, it is assumed that all frames sizes used to compute these tables are increased by the length of the fixed-size header. This header can also be used to transmit a packet ID field to support the data multiplexing operation described in Section 2.2.1.

difference between the actual and expected buffer fullness values. When the server receives one of these feedback messages, it will add extra delay to the transmission schedule in order to compensate for the error.

Since the Transmission Schedule Table computed by the server was constructed to prevent starvation by assuming that the receiver's clock was running at its fastest possible pace, the value of the actual buffer fullness will always be greater than or equal to the value of the expected buffer fullness. Therefore, the receiver must allocate extra buffer space beyond the minimum buffer size which was assumed for the case of an ideal network environment as discussed in Chapter 3.

In order to determine how often a feedback message should be sent to the server, it is necessary to bound the time for a feedback message to be transmitted and processed by the server, as well as the tolerance of the set-top box's local clock. The following notation will be used to represent these bounds:

Assume that:

1. The time for a feedback message to be transmitted to and processed by the server is bounded by t_B (measured in seconds).
2. The clock error tolerance of each receiver is bounded by E_R . (measured in parts per million).
3. The transmission rate is r (measured in bits/second).

Therefore, within the time that it takes for a feedback message to be transmitted and

processed by the server, $S = t_B \times \left(\frac{E_R}{1 \times 10^6} \right) \times r/8$ bytes of additional data (beyond that expected for the ideal case) may arrive at the receiver. Thus, it is required that the receiver allocate additional buffer space and transmit a feedback message to the server before the amount of room left in this buffer exceeds S .

4.3 Protocol Refinements for Intermedia Synchronization

This section extends the Continuous Media Transport Protocol developed in Chapter 3 in order to provide a mechanism for synchronizing the play-out of two or more continuous media streams.

Under the protocol refinement described in Section 4.1, play-back of a single continuous media stream in isolation requires that a set-top box:

1. monitor its receive buffer until B_{SU} bytes have been received,
2. wait an amount of time equal to J_{MAY} , where J_{MAY} represents the jitter bound of the network, and
3. begin play-back.

However, synchronized play-back of multiple continuous media streams requires that the playback of each stream begin at the same time. Therefore, a set-top box cannot begin playback of any stream until the receive buffer for each stream has prefilled by the proper amount.

The amount of time, t_{SU} , required to prefill a stream's receive buffer by the proper

amount is determined by the transmission schedule for that stream, and thus will vary with each stream. In order to allow for intermedia synchronization, the continuous media server will delay starting the transmission of some streams in order that the buffer fullness values of all streams reach their required point at the same instant in time. For example, if an audio and a video stream are to be transmitted, with $t_{su} = 10$ msec for the audio stream and $t_{sv} = 8$ msec for the video stream, then transmission of the audio stream will begin $10 \text{ msec} - 2 \text{ msec} = 8 \text{ msec}$ before transmission of the video stream.

CHAPTER 5. CONCLUSION

The protocol developed in this dissertation has been designed to meet the real-time transmission requirements of continuous media data streams. The network architectures most likely to be used for wide-area video on demand distribution were examined in order to formulate a list of requirements which the protocol should meet. These requirements included:

1. the ability to meet the real-time play-out deadlines at a receiver without overflowing the receiver's available buffer space,
2. the ability to operate correctly in networks which may introduce jitter into the packet delivery times,
3. the ability to perform periodic resynchronization in order to correct for loss of synchronization due to non-zero clock tolerances, and
4. the ability to playback multiple related continuous media streams in a synchronized manner at a set-top box.

The Continuous Media Transport Protocol has been designed to meet these requirements. In addition, the protocol has been designed to minimize the buffer requirements at a set-top box. This optimization can result in significant cost savings for a practical system.

Several areas of future work are possible. Further extensions to the protocol could be added to support operations such as random access to arbitrary points in a continuous media data stream, pause and resume operations, and possibly fast-forward and reverse playback modes. In addition the protocol could be extended to provide synchronized playback in a

multicast or broadcast environment.

BIBLIOGRAPHY

- Allen, James R., Blaise L. Heltai, Arthur H. Koenig, Donald F. Snow, and James R. Watson. 1993. VCTV: A Video-On-Demand Market Test. AT&T Technical Journal 72, no. 1 (January/February): 7–14.
- Anderson, David P., and George Homsy. 1991. A Continuous Media I/O Server and Its Synchronization Mechanism. IEEE Computer 24, no. 10 (October): 51–57.
- Burpee, David S., and Paul W. Shumate Jr. Emerging Residential Broadband Telecommunications. Proceedings of the IEEE 82, no. 4 (April): 604–614.
- Campbell, Andrew, Geoff Coulson, Francisco Garcia, and David Hutchison. 1992. A Continuous Media Transport and Orchestration Service. Computer Communication Review 22, no. 4 (October): 99–110.
- Chang, Yee-Hsiang, David Coggins, Daniel Pitt, David Skellern, Manu Thapar, and Chandra Venkatraman. 1994. An Open-Systems Approach to Video on Demand. IEEE Communications Magazine 32, no. 5 (May): 68–80.
- Ferrari, Domenico. 1990. Client Requirements for Real-Time Communication Services. IEEE Communications Magazine 28, no. 11 (November): 65–72.
- _____. 1992. Delay Jitter Control Scheme for Packet-Switching Internetworks. Computer Communications 15, no. 6 (Ju.y/August): 367–373.
- Ferrari, Domenico, and Dinesh C. Verma. 1990. A Scheme for Real-Time Channel Establishment in Wide-Area Networks. IEEE Journal on Selected Areas in Communications 8, no. 3 (April): 368–379.
- Ganger, Gregory R, Bruce L. Worthington, Robert Y. Hou, and Yale N. Patt. 1994. Disk Arrays High-Performance, High-Reliability Storage Subsystems. IEEE Computer 27, no. 3 (March): 30–36.
- ISO/IEC JTC 1/SC29. 1993. Final Text for ISO/IEC 11172-1, Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s – Part 1: Systems. Geneve: ISO/IEC.
- _____. 1993. Final Text for ISO/IEC 11172-1, Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s – Part 2: Video. Geneve: ISO/IEC.

- _____. 1993. Final Text for ISO/IEC 11172-1, Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s – Part 3: Audio. Geneva: ISO/IEC.
- Karlsson, Gunnar, and Martin Vetterli. 1989. Packet Video and Its Integration into the Network Architecture. IEEE Journal on Selected Areas in Communications 7, no. 5 (June) 739–751.
- Kurose, Jim. 1993. Open Issues and Challenges in Providing Quality of Service Guarantees in High-Speed Networks. Computer Communication Review 23, no. 1 (January) 6–15.
- Le Gall, Didier J. 1991. MPEG: A Video Compression Standard for Multimedia Applications. Communications of the ACM 34, no. 4 (April): 46–58.
- _____. 1992. The MPEG Video Compression Algorithm. Signal Processing: Image Communication 4, no. 2 (April): 129–140.
- Little, Thomas D. C., and Arif Ghafoor. 1990. Synchronization and Storage Models for Multimedia Objects. IEEE Journal on Selected Areas in Communications 8, no. 3 (April): 413–427.
- Little, Thomas D. C., and Arif Ghafoor. 1991. Multimedia Synchronization Protocols for Broadband Integrated Services. IEEE Journal on Selected Areas in Communications 9, no. 9 (December): 1368–1382.
- Little, Thomas D. C., and Arif Ghafoor. 1992. Scheduling of Bandwidth-Constrained Multimedia Traffic. Computer Communications 15, no. 6 (Ju.y/August): 381–387.
- Little, Thomas D. C., and Dinesh Venkatesh. 1994. Prospects for Interactive Video-on-Demand. IEEE Multimedia 1, no. 3 (Fall): 14–23.
- Loeb, Shoshana. 1992. Delivering Interactive Multimedia Documents over Networks. IEEE Communications Magazine 30, no. 5 (May): 52–59.
- Lougher, P., and D. Shepherd. 1993. The Design of a Storage Server for Continuous Media. The Computer Journal 36, no. 1: 32–42.
- Luther, Arch C. 1989. Digital Video in the PC Environment. New York: Intertex Publications, Macgraw-Hill Book Comapny.
- MacInnis, Alexander G. 1992. The MPEG Systems Coding Specification. Signal Processing: Image Communication 4, no. 2 (April): 153–159.

- Miller, Matthew D. 1994. A Scenario for the Deployment of Interactive Multimedia Cable Television Systems in the United States in the 1990's. Proceedings of the IEEE 82, no. 4 (April): 585–589.
- Montgomery, Warren A. 1983. Techniques for Packet Voice Synchronization. IEEE Journal on Selected Areas in Communications 1, no. 6 (December): 1022–1028.
- Ngoh, L. H., and T. P. Hopkins. 1989. Transport Protocol Requirements for Distributed Multimedia Information Systems. The Computer Journal 32, no. 3: 252–261.
- Nicolaou, Cosmos. 1990. An Architecture for Real-Time Multimedia Communication Systems. IEEE Journal on Selected Areas in Communications 8, no. 3 (April): 391–400.
- Ramanathan, Srinivas, and P. Venkat Rangan. 1993. Feedback Techniques for Intra-Media Continuity and Inter-Media Synchronization in Distributed Multimedia Systems. The Computer Journal 36, no. 1: 19–31.
- Ramanathan, Srinivas, and P. Venkat Rangan. 1993. Adaptive Feedback Techniques for Synchronized Multimedia Retrieval over Integrated Networks. IEEE/ACM Transactions on Networking 1, no. 2: 246–260.
- Rangan, P. Venkat. 1992. Designing an On-Demand Multimedia Service. IEEE Communications Magazine 30, no. 7 (July): 56–54.
- Rangan, P. Venkat, Srinivas Ramanathan, Harrick M. Vin and Thomas Kaepfner. 1993. Techniques for Multimedia Synchronization in Network File Systems. Computer Communications 16, no. 3 (March): 168–176.
- Rangan, P. Venkat, and Harrick M. Vin. Designing File Systems for Digital Video and Audio.
- Rangarajan, Aravind, Glenn L. Cash, Donald L. Duttweiler, Hsueh-Ming Hang, Barry G. Haskell, and Atul Puri. 1993. Image and Video Coding Standards., AT&T Technical Journal 72, no. 1 (January/February): 67–89.
- Ravindran, K. and Vivek Bansal. 1993. Delay Compensation Protocols for Synchronoization of Multimedia Data Streams., IEEE Transactions on Knowledge and Data Engineering 5, no. 4 (August): 574–589.
- Reddy, A. L. Narasimha, and James C. Wyllie. 1994. I/O issues in a Multimedia System. IEEE Computer 27, no. 3 (March): 69–74.
- Rosenberg, Jonathan, Gil Cruz, and Thomas Judd. 1992. Presenting Multimedia Documents over a Digital Network. Computer Communications 15, no. 6 (July/August): 374–380.

- Ruemmler, Chris, and John Wilkes. 1993. Modelling Disks. HP Laboratories Technical Report HPL-93-68.
- Quinnell, Richard A. Building the Digital Video Pipeline. 1994. EDN 22 December, 39–42.
- Steinmetz, Ralf. Synchronization Properties in Multimedia Systems. 1990. IEEE Journal on Selected Areas in Communications 8, no. 3 (April): 401–412.
- Steinmetz, Ralf, and J. Christian Fritzsche. 1992. Abstractions for Continuous Media Programming. Computer Communications 15, no. 6 (July/August): 396–402.
- Vin, Harrick M, and P. Venkat Rangan. 1993. Designing a Multi-User HDTV Storage Server. IEEE Journal on Selected Areas in Communications 11, no. 1 (January): 153–164.
- Wolfinger, Bernd, and Mark Moran. 1991. A Continuous Media Data Transport Service and Protocol for Real-Time Communication in High Speed Networks. University of California, Berkeley.
- Woo, Miae, Naveed U. Qazi, and Arif Ghafoor. 1994. A Synchronization Framework for Communication of Pre-orchestrated Multimedia Information. IEEE Network. 8, no. 1 (January/February): 52–61.