On-Demand Hypermedia/Multimedia Service over Broadband Networks*

C. Bouras[1,2] V. Kapoulas[1,2] D. Miras[1] V. Ouzounis[1]
P. Spirakis[1,2] A. Tatakis[1]

[1] Computer Engineering Dept.
Univ. of Patras
265 00 Patras, Greece

[2] Computer Technology Institute
Kolokotroni 3
262 21 Patras, Greece

Abstract

In this paper we present a unified approach for delivering hypermedia/multimedia objects over broadband networks. Documents are stored in various multimedia servers, while the inline data may reside in their own media servers, attached to the multimedia servers. The described service consists of several multimedia servers and a set of functions that intend to present to the end user interactive information in real-time. Users interact with the service requesting multimedia documents on demand. Various media streams are transmitted over different parallel connections according to their transmission requirements. The hypermedia documents are structured using a hypermedia markup language that keeps information of the spatiotemporal relationships among document's media components.

In order to deal with the variant network behavior, buffering manipulation mechanisms and grading of the transmitted media quality techniques are proposed to smooth presentation and synchronization anomalies.

1. Introduction

The recent years have introduced an essential development in several areas of computing technology; high speed networks (broadband ISDN, FDDI, ATM) are dominating, workstations are becoming more and more powerful, new media coding, compression and storage techniques are emerging. At the same time, multimedia technologies are becoming the edge, and new operating system aspects and protocols to deal with multimedia information are being proposed.

All these developments have significantly contributed to the emergence of innovative applications that make feasible the transmission of multimedia data over a network. Such applications have diverse requirements concerning the network, the workstations characteristics, the playout mechanisms, the multimedia generation and presentation methods, and may be applicable for distance education, telemedicine, multimedia news services, electronic magazines, remote access to virtual galleries, etc. These applications should be considered as services and may involve issues such as subscription, authentication, pricing and privacy.

A multimedia service may invoke several functional components throughout its operation; database systems, retrieval, transmission and buffering mechanisms, media coding techniques and presentation devices. Each of them requires specific manipulation and may impose presentational malfunction, thus, user dissatisfaction. When dealing with distributed environments, it is of major importance the accomplishment of synchronization among the different media that compose a multimedia object. Following, we identify some key issues concerning an on demand delivery of multimedia objects.

1. Multimedia objects or documents should be considered as a composition of different types of media (text, images, graphics, audio, video) that are appropriately placed in space and time to form a presentation scenario. In addition, in order to relate various multimedia objects and construct an information web, the notion of *hypermedia* is introduced. To effectively manage such presentations, several models for the integrated modeling of hypermedia documents have been pro-

*This work is supported by the European Commission - TELEMATICS APPLICATIONS PROGRAMME under the project TRENDS (ET - 1024) and the Greek Ministry of Education.

posed and adopted. *HTML* experiences wide acceptance, yet lacks of real-time dimensions in its presentation mechanisms. *HyTime* and *MHEG* are becoming standards. The *Dexter Reference Model* [2] provides a basis for interchange and interoperability standards. The *Amsterdam Hypermedia Model* [3] extends the Dexter Model, featuring notions of time, high level presentation and link context. Furthermore, several methods have been proposed for the modeling of storage, retrieval and synchronization of distributed media streams [5, 6, 7, 11]. In [10] general issues on synchronization are discussed.

2. Different media streams that participate in a multimedia presentation have diverse transmission requirements concerning the bandwidth, the tolerance to network's delays, the error rates. Upon connection to the service, a negotiation process should be invoked to reserve the required network parameters. This negotiation process should take into account the user's desirable *Quality of Service (QoS)* parameters and the network's condition.

3. The existence at the receiving end of a buffering mechanism is essential in order to provide temporary storage for the incoming data before their presentation, as well as to smooth presentation anomalies due to network's load conditions and probabilistic behavior. Several techniques to handle such buffering schemes in conjunction with the synchronization problem among various media have been suggested, involving buffer's sizes and mechanisms that should be activated on buffer's overflow or underflow conditions, e.g. [8].

In the rest of the paper, we discuss our approach to the aforementioned issues, and we present a design schema for applications that acquire real-time characteristics and require media synchronization support.

2. Service Overview

A multimedia/hypermedia service should be considered as a set of functions that intend to present to the end user interactive information that involves a variety of media. A minimal set of such functions that addresses the user involvement should be:

- the subscription to the service as well as authentication and pricing primitives,
- the determination of the user's desired presentation parameters (e.g. the audio or video quality that the user desires),
- the retrieval of a document or a topic of interest,
- the navigation through different hypermedia documents,
- the adjustment of the synchronization and presentation options.

The service consists of a set of multimedia servers distributed over a broadband network. Hypermedia documents are stored in these servers. Each multimedia server may consist of various media servers (image servers, audio servers, etc.). The internal structural presentation of a hypermedia object is stored in a multimedia server, while the inline data that compose the document may reside on their own media servers attached to the multimedia server.

Multimedia documents can be linked via hyperlinks to provide a logical interconnection among related documents. Linked documents may be stored in the same or in different servers.

Requested documents, as well as their media data are transmitted to the connected user's workstation, where specific processes are invoked to handle, buffer and present/play the incoming streams.

The presented design approach addresses the above characteristics, specifying in a transparent way how hypermedia documents are organized and stored, how interactions and data delivery are managed, and how the presentation of the hypermedia documents is performed.

3. Structuring Hypermedia Objects

In order to effectively store, retrieve, represent and interchange hypermedia objects among distributed places, and also regain their original spatial and temporal presentation scenario at the receiving edge, a reference model should exist to meet the above needs. Such a model should feature:

- media content encapsulation in the integrated context (e.g. a text),
- association between media content and its presentational attributes (e.g. text's fonts, presentation's background or foreground colors, image's dimensions),
- spatio-temporal internal representation of a document's entities, that also implies the synchronization among related in time media,
- linking notions among the various hypermedia documents or among the related inline media; the clue feature of the hypermedia abstraction.

Our model is divided into four logical abstractions, namely the *content*, the *layout*, the *synchronization* and the *interconnection*.

The content refers to the inline media entities and specifies their characteristics, such as, where they are stored, the encoding standard that is used, etc. The layout consists of a set of rules that internally specify how the different media will be presented on the user's desktop. The synchronization specifies the temporal (time) relationships among the various media. The interconnection provides a method for connecting hypermedia documents with other ones. This is achieved using *hyperlinks*. Hyperlinks may be divided in two categories, namely *sequential* and *explorational*. Sequential links indicate the document that should be followed by the user in order to preserve the logical sequence (or the author's sequence) of the selected information topic. Explorational links can be used to override the logical sequence and to provide access to related information.

We have designed a hypermedia markup language that is influenced by HTML and offers a set of primitives for the presentation and synchronization of the inline media and the interconnection among documents. A document is a composition of different media that are appropriately placed in time and space to form a playout scenario. Every single media (text, image, graphics, audio, video) that takes place in the formation of the playout scenario, is described by the markup language. Each media has its static characteristics, such as the playout or the content structure, and its time characteristics, such as, the relative time this particular media is played out according to the presentation scenario, the playout duration, etc. The representation of a document by the markup language is actually a text file. Several *tags* and *keywords* are used to denote a specific meaning or operation. The conceptual ideas for representing the layout structure are borrowed by HTML. The tags and the keywords are used to indicate a specific form of a document, such as, indication of media type (text, image, etc.), media placement and annotation, paragraph structuring, text alignment, headings and other presentation primitives.

Despite HTML's wide acceptance, its big disadvantage is the lack of synchronization primitives for media with time dependent characteristics (e.g. audio, video). We try to overcome this problem by adding to the markup language time features. Issues on the proposed markup language are discussed in [1].

The key point is the introduction of the "media relative start time". The different media have a specific presentation start time, relative to the presentation start time of the whole document, and a duration. Thus, by allowing each media to have its own start time and embedding this feature in the markup language, we gain a simple way for representing media in time. A simple synchronization can be achieved using this "start time" feature.

Using this "startime" indication of each media that participates in a scenario, and taking into account its presentation duration, a playout scheduler at the client side, invokes a concurrent playout process to present the media data on the proper device before their deadline, as it is dictated by the "startime" attribute.

4. The Design Approach

Our design effort is concentrated on four major issues concerning a hypermedia/multimedia service. These four issues are:

- the management of user's requests for viewing documents,
- the manipulation of media streams transmission,
- the buffering and synchronization of various media,
- the presentation of the hypermedia document.

In Figure 1, the design approach is depicted.

A first issue that arises when dealing with such services, is the invocation, upon connection request, of a connection establishment mechanism in order to provide access to the service. This mechanism evaluates a set of parameters concerning the network and the connection's request options, to decide on connection admission or rejection. Such parameters are the network's condition the specific time the request is sent (e.g. network load, available bandwidth) and the potential load that will be caused due to the new connection. The load that a new connection introduces, is a combination of the resource requirements the data that should be transmitted holds (e.g. bandwidth, interarrival delay, delay jitter, packet loss probability), and the lower thresholds in QoS and Quality of Presentation the user is willing to accept.

The above parameters are evaluated in conjunction with the pricing contract of the specific user (a user who pays more should be serviced, even though it affects the other users), and the connection is either initiated or rejected.

Upon connection, the multimedia server is activated to retrieve the requested hypermedia document. The server sends the presentation scenario, that is actually the hypermedia markup language representation of the document, to the *presentation scheduler* at client's side.

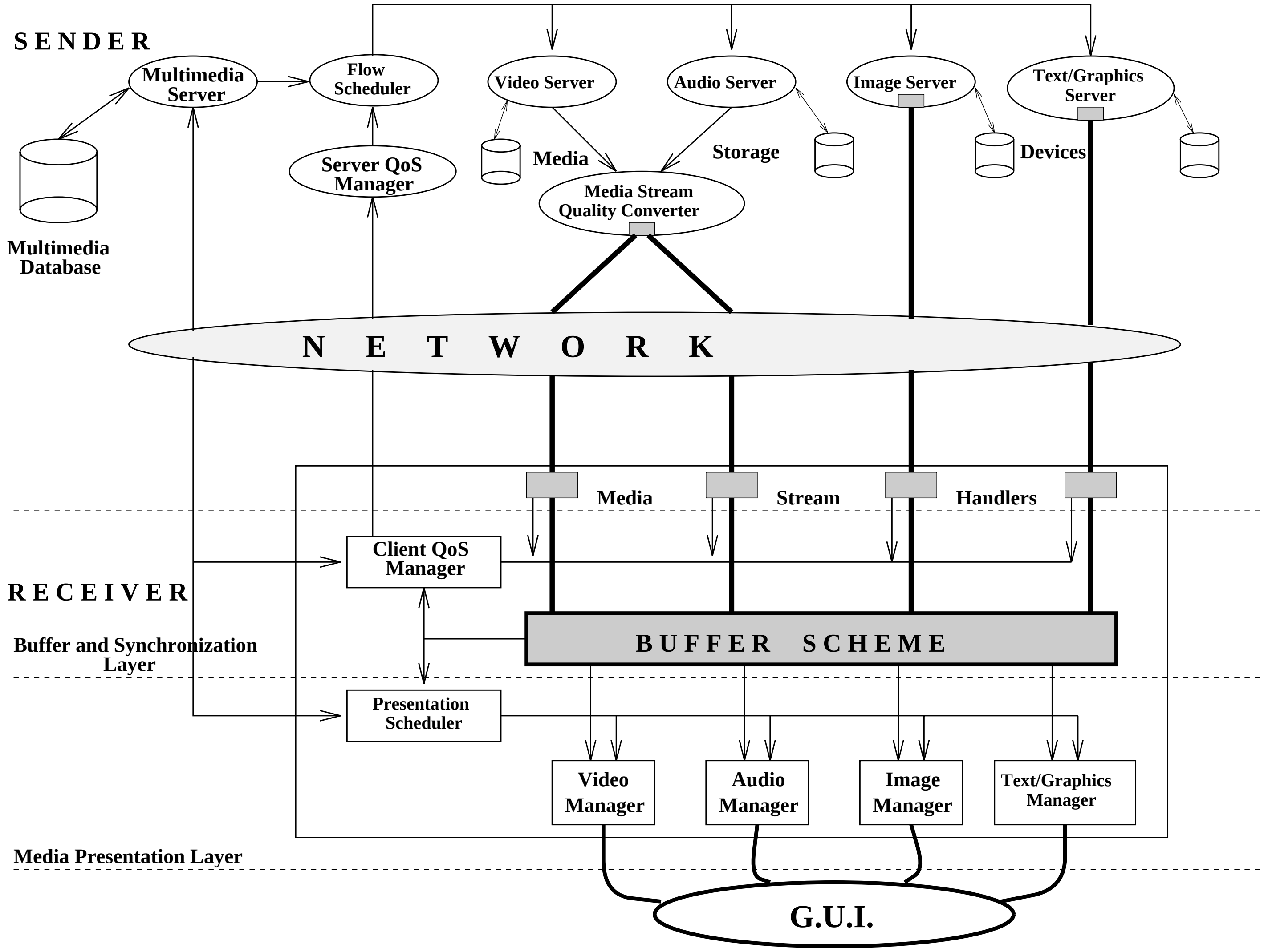

Figure 1. The Design Schema

The presentation scenario, as previously mentioned, specifies the spatiotemporal relationships among the media objects that compose the requested document. The presentation scheduler, by processing the presentation scenario, determines what media streams participate in the multimedia scenario, and when they should be invoked. This triggers the initialization of the corresponding media stream handlers, the associated buffer handlers, and the appropriate media presentation handlers. In addition, the presentation scheduler is responsible for the preservation and satisfaction of the time and presentation constraints of involved and related media, thus responsible for the *inter-* and *intra-media synchronization.*

At the server's site, the *flow scheduler* uses the retrieved from the *multimedia database* presentation scenario to compute a *flow scenario* for each participated media stream. This flow scenario specifies the sending start time instances of the corresponding media streams, as well as other transmission properties (e.g. transmission rates). Furthermore, it activates the appropriate media servers.

In order to deliver the various media data, each activated media server should set up the appropriate network connections. Each of these connections may have its own characteristics, based on the QoS requirements of the sending stream. Setting up a network connection, may be a static task (meaning that once its QoS parameters are determined upon establishment of the connection, they remain the same throughout its life) or dynamically adjustable; such a protocol is described in [4].

Network connections are experiencing significant delays, delay variation, and data loss in times of network congestion, which may cause undesirable disruptions in the presentation, affecting especially the time sensitive streams, like audio and video, and lead to gaps in audio/voice, jerky video and/or synchronization failure among related streams.

To partially face this problem, we introduce in our design schema a QoS manager at both the sending and receiving edges. Incoming data packets of a specific stream, besides other information, carry a timestamping indication which is used by the *Client QoS Manager* to carry out conclusions about the connection's condition, e.g. the packet delay, the delay jitter. Based on this information, the client QoS manager, periodically or in specifically calculated intervals, sends feedback reports to the sending side, the *Server QoS Manager*. Using such feedback reports, the service's server possesses knowledge of the overall network performance parameters, and accordingly takes corrective actions to improve service's operation. Flow scheduler identifies the specific media streams that are not transmitted as desired, and in cooperation with the corresponding *Media Stream Quality Converter* gracefully degrades (upgrades) the stream's quality, e.g. by increasing (decreasing) video compression factor or decreasing (increasing) audio sampling frequency. This results to less network traffic, thus more available bandwidth.

When dealing with the transmission of audio and video streams that should be synchronized, the service first applies the grading technique to the video stream, since audio or voice is considered to be more important to users, meaning that users can tolerate lower video quality than "not hear well". Besides, video streams have been turned out to be much more bandwidth consuming. Degrading media quality may be done down to several thresholds, taking into account at the same time and the user's desired levels of presentation quality, as have been expressed during the connection request. When falling to the lower threshold, the service may choose to stop transmitting the specific stream. The service should gracefully upgrade the media quality, when network's condition permits it. With the above described media quality grading technique, we provide the service with a mechanism for *long term* recovery support to synchronization anomalies.

In addition to the above described media degradation mechanism, the existence of a buffering scheme is needed to provide additional synchronization support. The recognition of the need of such a buffering scheme is constituted of the following reasons:

- a place should exist for the temporary storage of the incoming data, before they are forwarded to be played/presented,
- this amount of temporarily stored data is used to coarsely smooth the variations of the receiving rates of the various data streams, caused by network probabilistic behavior,
- furthermore, special mechanisms may be applied to deal with synchronization disruptions among related media. Such a mechanism that involves the monitoring of buffer occupancy with several actions concerning the synchronization of the corresponding media streams, using *dropping* and *duplication* is outlined in [8].

The described buffer, is a multiple thread queue; each thread is initialized after the establishment of its corresponding media connection.

One basic concept of the buffering layer, is that after the establishment of the parallel media connections, there is a relative delay in the presentation start time of the requested hypermedia object. This initial delay is inserted on purpose in order to feed each involved media buffer with an amount of data. This amount is statistically calculated at buffer's setup time, and depends on the specific transmitted media stream characteristics (frame/sample size, transmission rates, media encoding properties, tolerance to network delays). This length of each media buffer corresponds to a playback time, and we call this time interval, *media time window*.

The media time window is primarily used to smooth delays inserted by the network, the operating system, the transmission/receiving mechanisms. In this way, the experienced delays on data arrival first affect (decrease) the specific media time window (buffer's length) before affecting the quality of presentation and synchronization.

Furthermore, introduced data arrival delay variations and data loss lead to intermedia synchronization corruption, and a combinational algorithm, that involves buffer's occupancy levels and the presentation scheduler, should be applied to provide synchronization recovery. Transmitted frames/blocks have a presentation deadline but additionally, distinct frames/blocks that belong to related (synchronized) streams have timing constraints corresponding to their playout times with respect to each other.

Media synchronization may be categorized in

- *intramedia synchronization,* and
- *intermedia synchronization.*

Intramedia synchronization is satisfied when every media object (frame/sample/block) is available (delivered) for playing within its playout deadline. Intermedia synchronization is referred to related, due to a presentation scenario, media objects, and is achieved if the temporal requirements among these objects are met, on existence of timely delivery. *Intermedia skew* refers to the difference of the arrival times among media objects that should be synchronized.

When the buffer monitoring mechanism experiences buffer underflow, the presentation scheduler may lead to frame duplication in order to avoid noticeable gaps in presentation. Correspondingly, when buffer's occupancy exceeds some upper threshold, the scheduler should drop frames to decrease buffer's data. If intermedia skew is introduced among synchronized steams, that is caused by the buffer's underflow or overflow conditions, the scheduler may drop frames from the stream that leads in time or duplicate frames of the lagging stream in order to maintain a better synchronization. In this way, a *short term* synchronization incoherence recovery method is provided, before the long term synchronization support mechanism in the sending side is activated to provide media encoding grading.

5. Application's Functional Description

In the sequel, we describe the actions concerning the service's application protocol, that are the changes in the service's states that occur due to the various users and/or service's interactions/responses.

Initially, the user requests to connect to an existing server. An authentication primitive is invoked at the server side, in order to check that the user has the right to access the service. If the user is not a member of the service, the application prompts the user to fill in a subscription form. This form contains personal data such as name and address, telephone, e-mail, etc. By transmitting the form to the service's server, the user accepts the pricing policy and information content privileges. This form is transmitted to every server of the service, and a database entry of authorized users is updated while the pricing mechanism is initialized. The user may now access the contents of the service.

After the subscription primitive is invoked, the list of available topics (contents of the service) is sent by the connected server. From now on, the user can access the service requesting topics of interest. These topics can be stored in various multimedia servers. When the user requests a topic, this request is forwarded to the particular server where it is stored. For every associated document, the server where this document is stored is specified. The first document of the selected topic is transmitted and presented to the client's desktop. From the application point of view, sequential and explorational links are managed in the same way. If the requested document is stored in another multimedia server, a suspend connection primitive is invoked and a request for a new connection with a new server is performed. The suspended connection remains active for a period of time, in case the user requests to view a previous selected document. When this interval is passed the connection closes and the attached client is informed about the event. At that time client has an active connection with a new multimedia server. The user can issue a disconnect request from the service, at any time. The pricing primitive is informed about the request and the connection closes. In Figure 2, all the above mentioned transitions are depicted.

Interactive operations can be triggered by the user during the presentation of a document. The user can pause the presentation of the document and this request is forwarded to the involved media servers in order to stop transmitting the corresponding stream data. The user can request to resume presentation from the point it was paused. Additionally, user can request to reload an already selected document or to disable the presentation of a particular media involved in the selected document. The user may also annotate the selected document with his own remarks.

6. Implementation Issues

Part of the proposed design approach has been followed for the implementation of a prototype hypermedia service for distance education purposes. The service is tested over a 100 Mbps FDDI ring (has also been tested on Ethernet), using TCP/IP. The server and client parts currently run on Unix workstations (the client application is now implementing for Windows PC) and the X Motif libraries have been used for the development of the User Interface and other implementation issues.

The prototype is used for the delivery of prestructured multimedia lessons to remote users. The lessons are internally presented using the above mentioned hypermedia markup language [1]. For the transmission of data, we make use of the *Real-time Transport Protocol (RTP)* [9]. The RTP is an intermediate protocol that provides an application with end-to-end functions for transmission of real time data, such as audio and video. RTP data packets contain, besides pure data, auxiliary information concerning the following:

- a timestamp, indicating the packet's transmission or data sampling time instance,
- packet sequencing information,

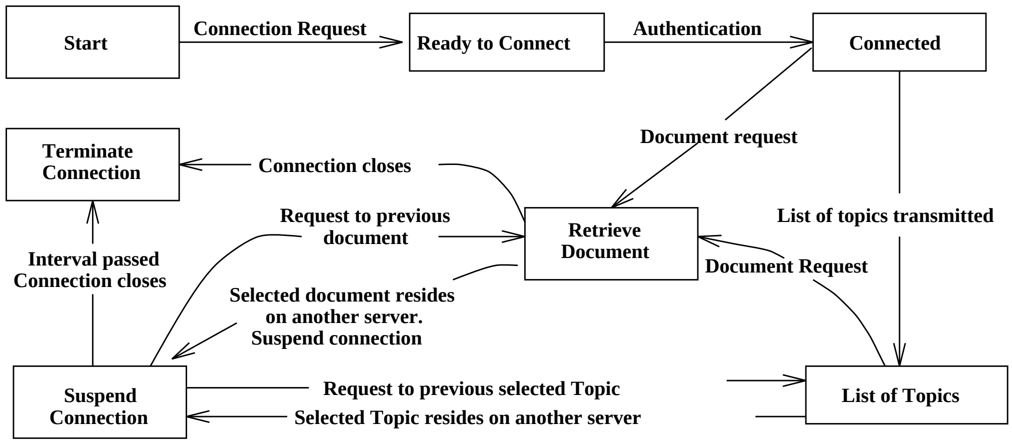

Figure 2. Application State Transition Diagram

- packet's data payload type (that is the coding format equivalent representation),
- other features.

RTP is followed by a control protocol *(Real-time Transport Control Protocol, RTCP)*, to support several actions, such as monitoring of the parameters of an ongoing session. RTP does not guarantee QoS in data transmission, and relies an network protocols (e.g. TCP) to make use of their transmission primitives.

We use this packet's header information to derive statistical measurements concerning network's parameters like packet's transmission delay, delay jitter and packet loss. RTCP feedback packets containing this kind of information/measurements are sent back to the sender, as *receiver's reports*. Using the timestamping information the packets carry, the media skew between the frame's/block's playout and real-time times can be calculated, thus buffer's monitoring mechanisms can be enabled, as described. We also use RTCP packets to realize the service's application protocol, as described in the previous section.

7. Conclusions and Future Work

The proposed design scheme introduces a method for delivering on demand hypermedia/multimedia documents over broadband networks. Several issues related to the media steams transmission, buffer and synchronization management and presentation are identified, and solutions are proposed. Especially, we described two closely related methods (long and short term) to deal with the synchronization disruptions in periods of network load.

We are currently working on the development of a general purpose distance learning service that actually relies on the described design scheme. A prototype has been implemented, tested and validated in various workstation environments (such as Windows PCs, SCO Unix and BSD Unix) and network platforms (such as Ethernet and FDDI).

Future work will focus on the improvement of the synchronization method used in conjunction with buffer's monitoring mechanisms, as well as the implementation of a testbed application on an ATM network. Special effort will be given in the further improvement of the hypermedia markup language to support more complicated presentational features.

8. Acknowledgments

The authors wish to thank the anonymous referees for their constructive comments on a previous version of this work. Thanks also to Mrs. Panagiota Fatourou and Mr. Dimitris Fotakis for their significant comments.

References

[1] C. Bouras, V. Kapoulas, D. Miras, V. Ouzounis, P. Spirakis, and A. Tatakis. A Framework for a Distributed Information Service Using Hypermedia/Multimedia Pre-Orchestrated Scenarios. Technical report, Computer Technology Institute, April 1996.

[2] F. Halasz and M. Schwartz. The Dexter Hypertext Reference. *Communications of the ACM*, 37(2):30–39, February 1994.

[3] L. Hardman, D. Bulterman, and G. Rossum. The Amsterdam Hypermedia Model: Adding. *Communications of the ACM*, 37(2):50–62, February 1994.

[4] A. Krishnamurthy and T. Little. Connection-Oriented Service Renegotiation for Scalable Video Delivery. In

Proc. of the 1st IEEE Intl. Conf. on Multimedia Computing and Systems (ICMCS '94), Boston, MA, volume 5, pages 502–507, May 1994.

[5] L. Li, A. Karmough, and N. Georganas. Synchronization in Real-Time Multimedia Delivery. In *Proc. IEEE ICC '92, Chicago, USA*, June 1992.

[6] T. Little and A. Ghaffoor. Synchronization and Storage Models for Multimedia Objects. *IEEE Journal on Selected Areas on Communications*, 8(3):413–427, April 1990.

[7] T. Little and A. Ghaffoor. Interval Based Conceptual Models for Time-Dependent Multimedia. *IEEE Transaction on Knowledge and Data Engineering*, 5(4):551–563, August 1993.

[8] T. Little and F. Kao. An Intermedia Skew Control System for Multimedia Data Presentation. In *Proc. 3rd Intl. Workshop on Network and Operation System Support for Digital Audio and Video, La Jolla, CA*, pages 121–132, November 1992.

[9] H. Schulzrinne and et. al. RTP: A Transport Protocol for Real-Time Applications, March 1995. Internet Draft, Internet Engineering Task Force.

[10] R. Steinmetz. Synchronization Properties in Multimedia Systems. *IEEE Journal on Selected Areas on Communications*, 8(3):401–412, April 1990.

[11] R. Steinmetz. Interval Based Conceptual Models for Time-Dependent Multimedia. *IEEE Transaction on Knowledge and Data Engineering*, 5(4):551–563, August 1993.