A FRAMEWORK FOR A DISTRIBUTED INFORMATION SERVICE USING HYPERMEDIA/MULTIMEDIA PRE-ORCHESTRATED SCENARIOS

C. BOURAS[1,2] V. KAPOULAS[1,2] D. MIRAS[1] V. OUZOUNIS[1]

[1] *Computer Engineering and Informatics Department, University of Patras, 26500 Rio, Greece*

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

e-mail: bouras@cti.gr

Abstract

Real time delivery of multimedia objects based on pre-orchestrated scenarios, seems to have several difficulties due to presentation deadlines that should be satisfied, and network's unpredictable behavior which may lead to synchronization anomalies among related media streams. Beyond these, a structural model that represents such scenarios is required. In this paper, we present a general framework that addresses the above mentioned issues, and we attempt to describe a unified approach for delivering hypermedia/multimedia objects over network connections preserving their timing constraints. We mainly concentrate on the study and development of a markup language that models the presentational structure of a multimedia object and provides primitives that assure playout synchronization of the different media streams that compose these hypermedia objects. Methods for keeping a constant quality level of presentation in times of network load are discussed.

Keywords: multimedia service, markup language, media synchronization, media buffers, QoS manager.

1 INTRODUCTION

The recent years we are experiencing an essential advent of the multimedia technologies and a rapid development of the network infrastructure. These developments make feasible the delivery of multimedia information among distant places over the network and contribute to the emergence of new applications that make use of multiple media.

Multimedia information systems are characterized by the need to compose and represent data of different types and formats for presentation, communication and storage. When dealing with distributed environments, essential is the need for synchronization among the various media streams that compose a multimedia object. We can categorize the several needs, problems and aspects of a distributed multimedia service as follows.

1. A flexible reference model for the storage and representation of multimedia scenarios should exist, that especially addresses issues such as retrieval policies and media synchronization, which express the distributed nature of multimedia data sources. Several methods and models have been suggested in the recent literature [1-8].
2. The existence of authoring tools is essential to the development of multimedia scenarios.
3. In order to access a multimedia service, a connection establishment mechanism should be provided. Such a mechanism should address issues like subscription or authentication to the service, determination of the pricing policy, adjustment of the *Quality of Service (QoS)* parameters.
4. To achieve a good presentation quality level, a buffering scheme should be applied to smooth the network's probabilistic behavior in time and make congestion conditions not easily noticeable by the user (see e.g. [9]).

In the rest of the paper, we describe our approach to the above issues, and we present a design of the proposed architecture of the service.

2 SERVICE ARCHITECTURE ISSUES

The proposed architecture (Figure 1) of the multimedia/hypermedia service should be considered as a set of functional modules that globally intend to present to the end user interactive information that involves a variety of media, such as formatted text, images, slides, graphs, animation, audio, video, etc. Different media are joined together to form multimedia objects. Users can navigate through these multimedia objects using special kind of links, called hyperlinks. Multimedia objects linked together via hyperlinks are usually called hypermedia objects or hypermedia documents. Hyperlinks may also join different types of data (text with text, an audio segment, an image with another image, etc.) that may be distributed over the network, forming an information web.

A user who accesses the service, may request to view a hypermedia document. Every hypermedia document has an associated spatial and temporal presentation scenario that is stored in the remote server. In the next section, we describe how this presentation scenario is modeled. Media, like text, figures, images, audio and video, that compose the hypermedia document, may reside in the same or another media server. Through the existence of multiple media servers, data retrieval and transfer are performed continuously in time, as the presentation scenario dictates. Time-dependent media objects, like audio and video, are stored digitized and compressed in a known format at their associated media server. Data retrieval and transmission is initiated by a connection establishment function between the client application and the server. The connection establishment function is driven by a scheduler which uses information incorporated in the presentation scenario. This information concerns the remote sources of the engaged with the scenario media streams. During the connection initialization, several actions to determine the parameters of the connection may take place, if the network infrastructure supports them.

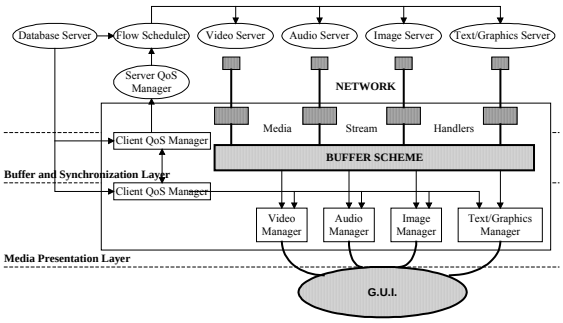

FIGURE 1: SERVICE ARCHITECTURE

In order to maintain an efficient service quality, feedback information, collected by the receiving side, about each connection's condition and presentation's quality, is periodically sent to the corresponding media servers. Such information refers to parameters like the end-to-end delay, delay jitter, packet loss, etc. These factors, may lead to presentation anomalies that are noticeable to the end user due to their probabilistic values in time. To partially overcome this problem, the sending source can gradually decrease the media's stream coding quality in times of network load. This results to less network traffic for the particular media stream. When applying this method to synchronized audio and video streams, it is preferable to decrease video quality first, because users seems to prefer to "hear well than see well". When available bandwidth increases, sending media's coding quality may analogously increase again.

In order to achieve the necessary grading of the media streams coding quality, special mechanisms may reside in server's side to provide operations for coding the retrieved data in a different format. Such mechanism may be hardware devices or software tools.

To deal with the variation of data transmission delays, a buffering mechanism is enabled in the client's side. The buffer is a multiple thread queue for temporarily storage of the incoming media data. Each buffer thread, or just buffer, is fed with an initial amount of data for each media stream right after the connection establishment and data transfer initialization. In this way, likely network delays in the future can be supported by the already queued data. This, of course, results to a decreasing buffer occupation. In a later section, we describe how the buffer's different levels of occupation are handled, and the relationship between buffer's size and media synchronization.

3 MODELING HYPERMEDIA DOCUMENTS

In order to interchange multimedia objects in a distributed environment and provide a multimedia application with the ability to regain the original spatio-temporal presentation scenario of the multimedia object's components, there is the need for the existence of a flexible and steady model that should address some basic features, like the following ones:

- Association between the media contents (e.g. text's content) and their presentational characteristics, like text attributes (fonts, size, bold, italics, paragraphs, separators, etc.), colors (background, foreground), and other.
- Placement in space and time (spatio-temporal presentation) of the several media data that compose a hypermedia/multimedia document.
- As a consequence, the synchronization in real time of media that should be presented/played together, following their presentation scenario.
- Linkage among the hypermedia objects or among the individual components of hypermedia documents, that drives the presentation through the various objects.

Several models have been proposed for the integrated modeling of multimedia documents [10-12]. We propose a model for structuring multimedia documents that faces the above mentioned features in a unified way. The proposed model is realized by the use of a hypermedia presentation markup language. This markup language was designed and prototypely implemented as part of the design and development process of our prototype application for the on-demand delivery of hypermedia documents.

Using the developed markup language for the modeling of hypermedia documents, we gain a flexible and interoperable, yet simple way to represent a pre-orchestrated multimedia scenario, especially under the existence of a companion authoring tool. This hypermedia markup language is influenced by *HTML (HyperText Markup Language)* and in a way, keeps its simplicity in constructing documents. A hypermedia document, as described by the model, is a text file which contains several *tags* and *keywords* to indicate the specific form of the document, such as, indication of media type (text, image, audio, etc.), media placement and annotation, paragraph structuring, text

form (fonts, size, alignment, headings, etc.) and other presentation options.

The basic principles and construction elements of the language are presented in more detail in [13, 14]. In the sequel, we show how a multimedia scenario is constructed using this markup language, and how a coherent playout presentation can be scheduled.

Assume the following hypermedia scenario: Hypermedia document contains a formatted text that is always shown throughout the presentation. At the start of the presentation (relative time) the image I_1 is shown having presentation duration d_{i1}. After the appearance of I_1, in the time instant t_{i2} after the presentation start, image I_2 is shown and kept on the display for duration d_{i2}. At time t_{a1} an audio segment A_1 should be heard synchronized with a video V of duration d_V. The two media should start and stop playing at the same time. In the time t_{a2} an audio segment A_2 plays out with duration d_{a2}. Figure 2 illustrates the graphical presentation of the scenario and its correspondence in term of playout timelines.

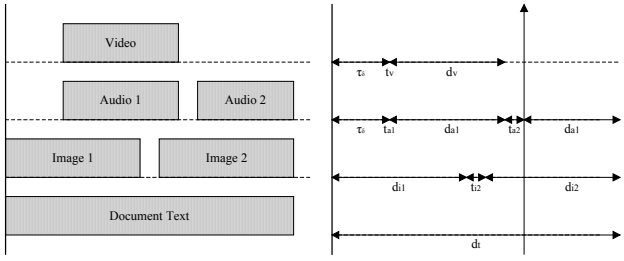

FIGURE 2: ILLUSTRATION OF A SIMPLE SCENARIO

Trying to transform the graphical notation of the scenario in terms of the described markup language, we result to the following part of language rules/expressions:

<TEXT> *text_body* </TEXT>
<IMG> SOURCE=I_1 ID=... STARTIME=0 DURATION=d_{I1} </IMG>
<IMG> SOURCE=I_2 ID=... STARTIME=t_{I2} DURATION=d_{I2} </IMG>
<AU> SOURCE=A_1 ID=... STARTIME=t_{A1} DURATION=d_{A1} </AU>
<AU> SOURCE=A_2 ID=... STARTIME=t_{A2} DURATION=d_{A2} </AU>
<VI> SOURCE=V ID=... STARTIME=t_V DURATION=d_V </VI>

At the client's side, in order to extract all the necessary timing information for the determination of the playout schedule, a preprocessing of the received presentation scenario is taken place. In this preprocessing, every media stream S_i is recognized by its corresponding language rule and a struct E_i is informed. This struct contains stream's S_i timing parameters like start time t_i and duration d_i, corresponding data position in the temporary storage mechanisms (media buffers), and other useful information. Acquiring this information, the playout scheduler process can arrange the presentation of each media stream according to its playout deadlines (that are expressed from the time instant t_i). For each media stream S_i, a concurrent presentation process is creating, to play S_i when its deadline expires. The playout algorithm can be described as below:

```
for i = 0 to num_of_structs_Ei
   create_a_playout_thread {a playout process}
   wait until current_relative_presentation_time = ti {wait until
      presentation time arrives}
   play incoming stream Si in nominal rate for duration d1
end_for
```

However, the activation of a hyperlink by the user in a time instant t_h will interrupt the presentation of the scenario, and trigger the operations for the presentation of the linked hypermedia document.

4 BUFFERING SCHEME

In the receiving edge of the service, the existence of a buffering mechanism is necessary. Incoming media data are temporally stored in multi-thread buffers. Every buffer thread has a different size according to the stored data characteristics. Media data are temporarily stored in the buffer to pass through a decoding procedure before being played. The main usage of a buffering scheme is to accommodate, using a statistical resource reservation policy or by just upgrading or degrading queued media quality, measured or negotiated connection's delay invariant behavior. Such variant delay may lead to synchronization or media presentation anomalies (jerky video, gaps in audio) due to buffer underflow.

The media time window is used to smooth delays inserted by the network, operating system and other factors. In this way, the experienced delays affect (decrease) the specific media time window (buffer's length) before affecting the quality of presentation and the synchronization. When the media time window decreases, some recovery actions may take place. Let p and q be the rates at which the buffer media is played and queued, respectively. Let also s be the queue size. When the queue size s reduces, which means that there is network congestion (thus buffer size comes near to underflow threshold, u), the playout rate p is controlled by duplicating frames to keep a balance between buffers size and the network load. Another solution is to reduce media presentation rate p. Conversely, if queue size increases, due to network congestion reduction, the playout returns to its normal operation (without duplicating packets). When buffer size increases to the overflow limit, the application may drop frames from playing and decrease the rates of requested data.

5 DESIGN SCHEMA AND FUNCTIONAL DESCRIPTION

Results and conclusions derived from previous sections analysis contributed to the design and implementation of a framework for a distributed information services that make use of the hypermedia notation. Users actions that involve request for hypermedia document are recognized by the user actions interpreter, and an application packet is transferred to the appropriate server of the service. By arrival to the server, the request packet is parsed and a query is transferred to the media database manager, that retrieves the requested data. Depending on network's condition, data can be transferred at the stored quality or can be decoded and coded again by dedicated devices, that reside to the corresponding media server, to decrease their bandwidth needs, thus the bandwidth load. Data are packetized, and sent to client. At client's side, packets are de-

coded and data are temporarily stored in their corresponding buffers. Queued data are manipulated by the presentation manager that holds hypermedia document's presentation scenario. Depending on data type, data are shown on the display by the Graphical User Interface (GUI) or played by appropriate devices (e.g. speakers).

Buffer manager constantly monitors buffer's occupation and in conjunction with the QoS monitor process extracts feedback information which is combined with information about connection parameters (delay, jitter, data loss) that are extracted from data packet headers in order to create the periodically receiver report. This report, contains statistical measurements concerning the delivery quality, and is sent to the media server. By examining this report information, the QoS manager of the server concludes whether it should degrade or upgrade the quality of data that are going to be delivered.

6 IMPLEMENTATIONS ISSUES

The proposed framework has been followed to the implementation of a prototype version of a hypermedia distributed application for educational purposes such as remote training and access to digital libraries. The prototype addresses most of the issues discussed in previous sections.

The lessons are hypermedia documents and are constructed using the hypermedia representation markup language described previously. To integrate the transfer of data and the application protocol, we use the *Real-time Transfer Protocol (RTP)* as a transport mechanism. RTP [15] is an Internet standard, and is used as an intermediate protocol that provides end-to-end transport functions suitable for applications transmitting real-time data such as audio and video. RTP is augmented by a control protocol (RTCP) to allow monitoring of the data delivery. We use RTCP to gain feedback information, useful for monitoring the connections parameters. RTP relies on other network protocols (e.g. TCP) to guarantee data transmission. RTP also provides mechanisms for timestamping and sequencing the transmitted packets, and indication of the transmitted data payload type, which is the coding format of the media object.

We use these mechanisms to gain intramedia and intermedia synchronization using the timing information they provide through packet timestamping, as well as calculating statistical measurements about network's parameters like the transmission delay, jitter and packet loss. We use RTCP to send this feedback information to media servers and to realize the service's application protocol.

Upon connection to the service, an authentication primitive may be invoked at the server's side to check whether the user has the right to access the service. If not, the user can subscribe to the service accepting the pricing policy.

7 CONCLUSIONS AND FUTURE WORK

We proposed a general framework that tries to identify and address all the necessary issues regarding an on demand delivery of orchestrated hypermedia documents. A hypermedia markup language, that models the spatio-temporal attributes of such multiple media constructions, is proposed. The hypermedia markup language combines the simplicity in use with the interoperability and extensibility of the model. Special policies were discussed to deal with network load conditions in order to maintain a stable presentation quality.

Future work includes the extension of the markup language to handle more complex presentation needs and will focus on the improvement of the synchronization method used as well as the implementation of a testbed application on broadband networks, such as ATM.

REFERENCES

[1] Cosmos Nicolaou, "An Architecture for Real-Time Multimedia Communication Systems", *IEEE Journal on Selected Areas in Communications*, Vol.. 8, No. 3, Apr. 1990.

[2] Thomas D.C. Little, Arif Ghafoor, "Interval-Based Conceptual Models for Time-Dependent Multimedia Data", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 5, No. 4, Aug. 1993.

[3] N. Hirzalla, B. Falchuk, A. Karmough, "A Temporal Model for Interactive Multimedia Scenarios", *IEEE Multimedia*, Fall 1995.

[4] P. Venkat Rangan, Harrick M. Vin, and Srinivas Ramanathan, "Designing An On-Demand Multimedia Service", *IEEE Communications Magazine*, Jul. 1992, pp. 56-64.

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

[6] D. Ferrari, "Client Requirements for Real-Time Communication Systems", *Network Working Group*, RFC 1193, Nov. 1990.

[7] Thomas D.C. Little, Arif Ghafoor, "Synchronization and Storage Models for Multimedia Objects", *IEEE Journal on Selected Areas in Communications*, Vol. 8 No. 3, Apr. 90.

[8] Yahya Y. Al-Salqan, "MEDIAWARE: On Distributed Multimedia Synchrronization", *CERC-TR-RN-95-007*.

[9] T.D.C. Little, F. Kao, "An Intermedia Skew Control System for Multimedia Data Presentation", *3rd International Workshop on Network and Operating System Support for Digital Audio and Video*, La Jolla, CA, Nov. 1992, pp. 121-132.

[10] F. Halasz, M. Schwarzt, "The Dexter Hypertext Reference", *CACM*, Feb. 1994, Vol. 37, No. 2.

[11] L. Hardman, D. Bulterman, G. Rossum, "The Amsterdam Hypermedia Model", *CACM*, Feb. 1994, Vol. 37, NO. 2.

[12] G. van Rossum, J. Jansen, K. S. Mullender, D.C.A. Bulterman, "CMIFed: A Presentation Environment for portable hypermedia documents", *In proceedings of the 1st ACM International Conference on Multimedia*, Anaheim, CA. Aug. 1993, pp. 183-188

[13] C. Bouras, V. Kapoulas, D. Miras, V. Ouzounis, P. Spirakis, A. Tatakis, "On-Demand Hypermedia/Multimedia Service over Broadband Networks", *5th IEEE International Symposium On High Performance Distributed Computing (HPDC-5)*, 6-9 August 1996, Syracuse, New York, USA, pp. 224-231

[14] C. Bouras, V. Kapoulas, P. Spirakis, A. Tatakis, "An HTML like language supporting time-depended transmission of hypermedia", *The Eighth International ACM Hypertext Conference-Hypertext 97*, Southampton, UK, April 6-11, 1997, (poster presentation)

[15] Ralf Steinmetz, "Synchronization Properties in Multimedia Systems", *IEEE Journal on Selected Areas in Communications*, Vol. 8 No 3, Apr. 90.