A Framework for Synchronous Delivery of Time-Dependent Multimedia Data[1]

T.D.C. Little

Multimedia Communications Laboratory
Department of Electrical, Computer and Systems Engineering
Boston University, Boston, Massachusetts 02215, USA
(617) 353-9877, (617) 353-6440 fax
tdcl@bu.edu

MCL Technical Report 04-15-1993

Abstract–Multimedia data often have time dependencies that must be satisfied at presentation time. To support a general-purpose multimedia information system, these timing relationships must be managed to provide utility to both the data presentation system and the multimedia author. Timing management encompasses specification, data representation, temporal access control, playout scheduling, and run-time intermedia synchronization. In this paper we describe the components of our framework for supporting time-dependent multimedia data encompassing these areas, and how they are assembled into a unified system.

Keywords: Time-dependent data, temporal modeling, multimedia databases, synchronization, scheduling.

[1]In *Multimedia Systems* Vol. 1, No. 2, 1993, pp. 87-94. This material is based upon work supported in part by the National Science Foundation under Grant No. IRI-9211165.

1 Introduction

Multimedia as a technology can now be interpreted to describe computer systems supporting audio and video as data types. Characteristic of these data types is the need for timely data retrieval and delivery in the provision of multimedia services. Such services require the ability to overcome delays caused by storage, communication, and computational components during data routing from their source to destination. To realize this general service, a number of disparate technical issues must be resolved. These issues include specification/authoring, data representation/organization, scheduling, and synchronization.

To support pre-orchestrated multimedia presentations, a means of creating the timing relationships among data elements is required. Such mechanisms must not restrict the creativity of a multimedia author, yet must lead to data storage organizations that are efficient and viable for supporting multimedia reproductions. To support the presentation of time-dependent data, scheduling disciplines associated with real-time operating systems are necessary. These data are either generated in real-time or are assumed to be retrieved and transmitted in real-time from storage. Therefore, specific scheduling is required for system components in the delivery path to the user: storage devices, the CPU, and the network. We also perceive the need for mechanisms to support *scalability* and *graceful degradation* of multimedia services for the system as a whole. Graceful degradation can be achieved by run-time control of multimedia sessions based on changing environmental conditions such as system loading. For example, digital audio and video service can be degraded to reduce system loading by dropping some video frames.

Consider the following example to illustrate these issues. Fig. 1 (adapted from [17]) shows a timeline representation of the components of a newscast called the "Action News." We assume that this application is pre-orchestrated and played-out on demand as typified by video-on-demand scenarios.

In this example a number of different data types are integrated to form a composite multimedia presentation. The management of the temporal component of this application encompasses its temporal orchestration, data representation for storage, human interaction during playback, and the resolution of real-time constraints during delivery to the user.

Recent work on the enabling technology for multimedia information systems abounds. Most work is specialized, dealing with focussed problems rather than broad solutions (e.g., specification: [6, 8, 13, 14, 15, 31], user interaction: [30, 32, 36], logical data representa-

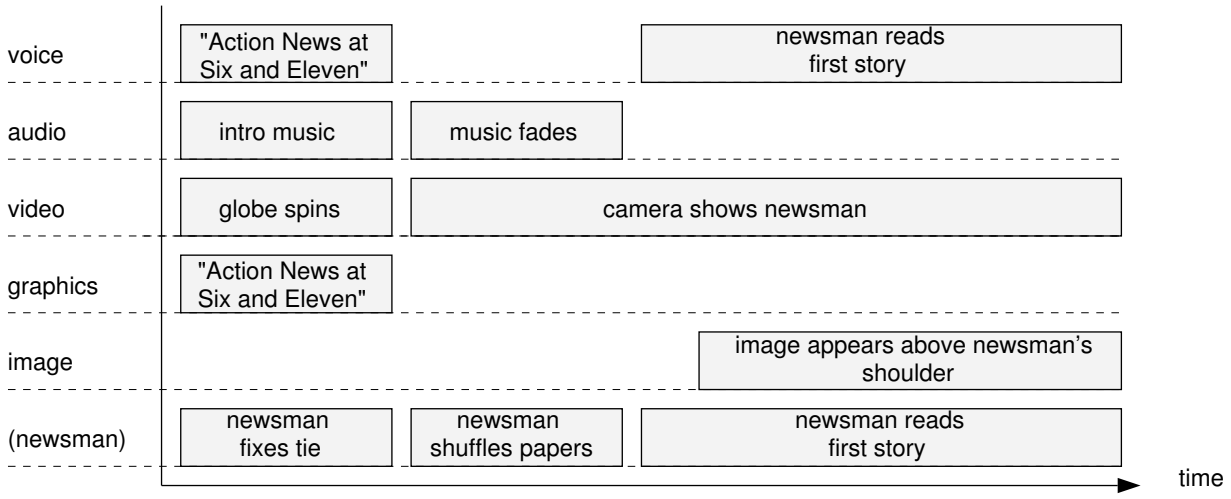

Figure 1: "Action News" Timeline Representation

tions: [2, 6, 22, 29], physical data organizations: [4, 5, 12, 28, 35], and system support: [1, 3, 7, 26, 27, 33, 34]). In this paper we describe an integration of solutions for overall timing management in a multimedia information system. This includes the identification of temporal relations between multimedia data objects, temporal conceptual database schema development, physical schema design, and access methods for synchronous retrieval.

The remainder of this paper is organized as follows. In Section 2 we introduce our framework for the management of time-dependent data and describe each component. Section 2.1 describes temporal specification and authoring issues. Section 2.2 summarizes data representations for storage of time-dependent data. Section 2.3 describes our scheduling approach based on the aforementioned data structures. In Section 2.4 we describe our run-time intermedia synchronization mechanism. Section 3 concludes the paper.

2 A Framework for Supporting Time-Dependent Data

We perceive a framework for managing time-dependent data as supporting many applications via fundamental services for multimedia data. Our framework is illustrated in Fig. 2. These services are designed to support aspects of time-dependent multimedia data presentation including temporal data management, temporal access control (e.g., fast-forward, reverse playout), playout scheduling, and fine-tuning of playout timing.

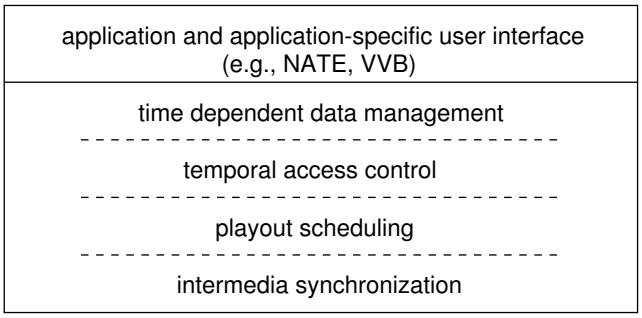

Figure 2: Framework for Supporting Time-Dependent Data

This organization is derived from our desire to support synchronous reproduction of multimedia presentations as created by an unrestricted authoring process. Each of these components and their interaction with other components is described in detail in the following subsections.

2.1 Timing Specification

Time-dependent multimedia data are characterized by their need to be presented in a timely, human-consumable form. Typically, timely delivery of these data is facilitated by dedicated synchronous mechanisms for playback (e.g., the electromechanical system comprising a VCR or a CD player). When this type of data is applied in more general-purpose computer data storage systems (e.g., disks), many interesting new capabilities are possible, including random access to the temporal data sequence and time-dependent playout of static data (e.g., animation). However, the generality of such a system eliminates the dedicated physical data paths and the implied data structures found in synchronous mechanisms and sequential storage.

To support a time-based representation, a multimedia system must capture the representation using an appropriate data structure that is suitable for subsequent application and user interaction functionality. In the following, we introduce our specification technique for time-dependent multimedia data that is necessary to support authoring and user interaction.

2.1.1 Specification of Timing – Authoring

The creation of time-dependent multimedia presentations requires some means of abstracting the author's intentions via a representational scheme. Both language-based approaches, including scripting, and flow graph and icon-based approaches have been proposed [2, 8, 13, 14, 15, 22, 31, 36]. In each case, one of the most significant requirements is the ability to represent concurrency and to specify real-time presentation timing. Graphical models have the advantage of pictorially illustrating synchronization semantics, and are suitable for the visual, icon-based orchestration of multimedia presentations. Suitable graph-based representations include timelines, flowgraphs, the Timed Petri net (TPN) [11, 22, 32], and temporal hierarchies [24, 29]. Fig. 3 represents a TPN for the "Action News" example of Fig. 1. This TPN model explicitly captures any of the temporal relations, and can provide simulation in both the forward and reverse directions. Each *place* in this TPN represents the playout of a multimedia object while transitions represent synchronization points. Other TPN models can be used in a similar manner to capture nondeterministic interactions caused by a user (e.g., browsing) [11, 32].

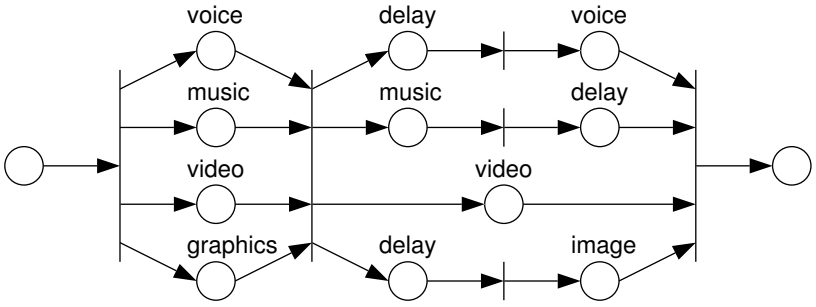

Figure 3: TPN for the "Action News" Example

Few of these language or graph-based representation techniques specify appropriate data structures to support subsequent temporal access control (TAC) operations in a general way when data are preorchestrated and maintained in a database. We use an approach that maps from a specification methodology to a database schema using the TPN and a relational database model [24, 22]. In this case, timing relationships are described by a timeline representation in an unstructured format (Fig. 1), or by a TPN in a structured format (Fig. 3).

2.1.2 Interaction and Time-Dependent Data

When a human user interacts with a multimedia system, the application must synchronize the user and the external world to the multimedia presentation. This can take the form of starting or stopping the playout of an object, posing queries against the database, browsing through objects, or other inherently unpredictable user or sensor-initiated activities. For continuous-media systems, user interaction also implies random access to a sequential form of information. These TAC operations include functions such as reverse, fast- forward, fast-backward, midpoint suspension, midpoint resumption, random access, looping, and browsing. TAC operations are feasible with existing technologies, however, when non-sequential storage, data compression, data distribution, and random communication delays are introduced, the provision of these capabilities can be very difficult.

Surrogate travel is an application that illustrates the use of TAC operations. For example, in Lippman's "movie map," a virtual "drive" down the streets of Aspen is achieved through animation of still images from a laserdisc [20]. This virtual "drive" allows "turns" and corresponding jumps out of the temporal-sequential nature of the sequence of images corresponding to a street. In addition to supporting TAC functionality, the surrogate travel application must be supported by an underlying data organization that provides branching options such as the aforementioned "turns."

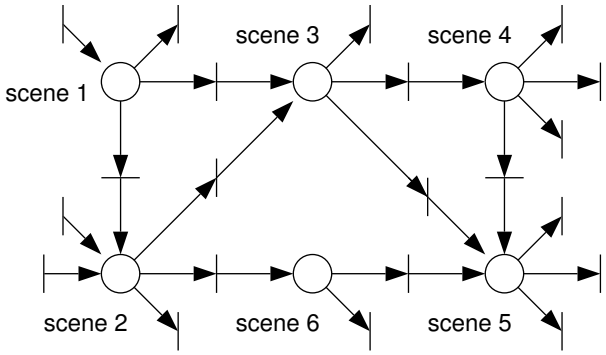

Figure 4: PNBH for Representing Relationships Among Scenes of a Motion Picture

It is possible to represent these temporal-sequential semantics by using a graph-based model such as the Petri net [32]. Such a Petri-Net-Based-Hypertext (PNBH) expresses information units as net places and links as net arcs. Transitions in PNBH indicate the traversal of links, or the browsing of information fragments. For example, in Fig. 4 we show a PNBH network consisting of segments of an interactive movie. These segments can be

played-out in a random order, as selected by the user and restricted by the semantics of the net. The fundamental difference between the PNBH and our TPN is the support for random user interaction [22]. Our TPN specifies exact presentation-time playout semantics which are ideal for real-time presentation scheduling. These two models complement each other for specifying both user interaction and orchestration.

2.2 Storage of Time-Dependent Data

In this section we describe our temporal data models for supporting TAC functionality and show how these models are applied in the framework for supporting time-dependent multimedia data.

2.2.1 Logical Data Structures

Temporal intervals can be used to model multimedia presentation by letting each interval represent the presentation or *playout* duration of some multimedia data element, such as a still image or an audio segment. The specification of timing for a set of intervals is achieved by indicating element durations for each data element and the relative timing among elments.

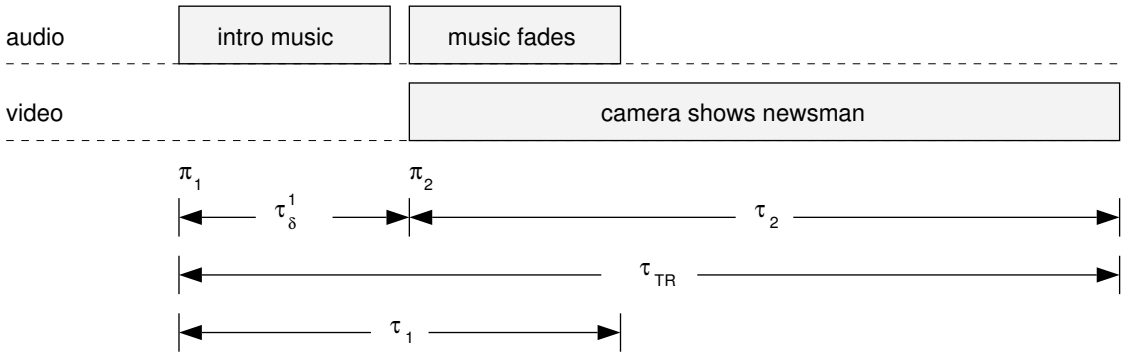

Figure 5: Illustration of the Timing Parameters for a Portion of the Action News

We have recently extended the generality of our initial temporal-interval-based (TIB) modeling approach with n-ary temporal relations [24]. Like the binary case, the n-ary temporal relation is characterized by a start time π_i, interval duration τ^i, and end time for a data element i. The relative positioning and time dependencies are captured by a delay τ^i_δ, as is the overall duration of a set of temporally related elements τ_{TR^n}. These parameters are illustrated in Fig. 5 for a subset of the "Action News" example. To capture

the semantics of the n-ary temporal relations and the object orchestration technique, we group multimedia objects and identify them with temporal parameters. This process is achieved by a hierarchical structure comprised of multimedia objects [24] which are assigned temporal attributes.

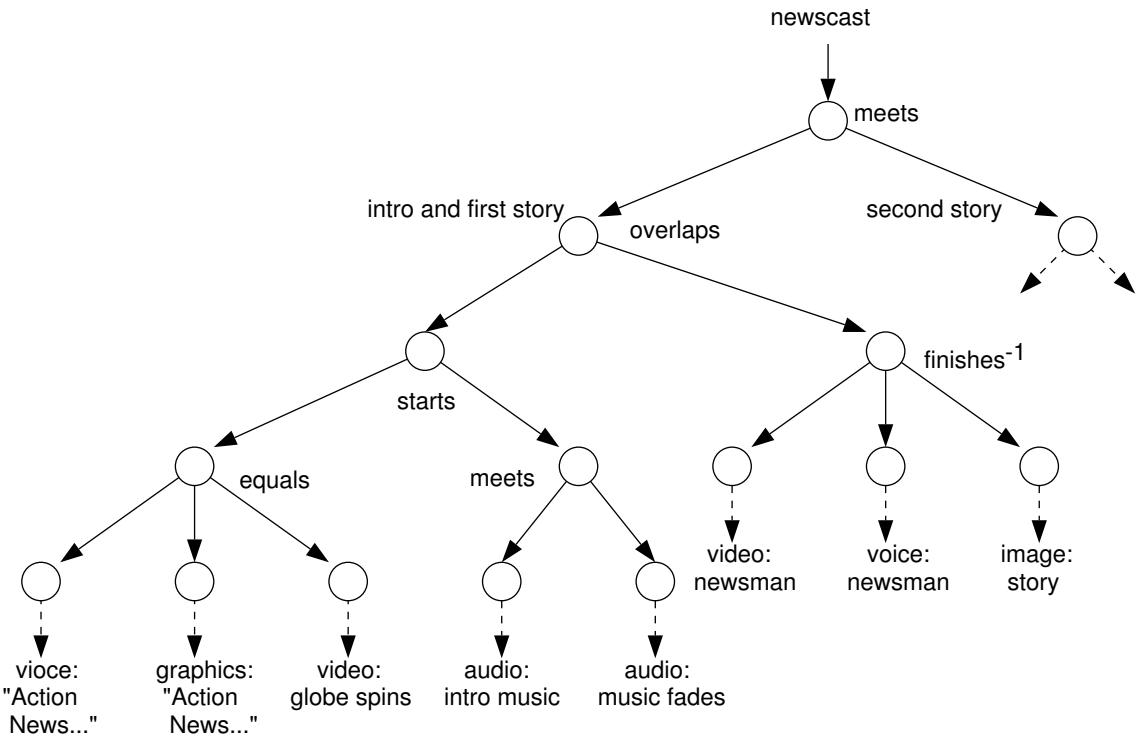

Figure 6: Temporal Hierarchy for the "Action News"

By using this approach, a timeline or TPN representation can be translated to a conceptual schema in the form of a temporal hierarchy representing the semantics of the TIB specification (Fig. 6). Subsets or subtrees of this hierarchy represent subsets of the specification, illustrating the capability of composing complex multimedia orchestration (e.g., first story or second story in the "Action News" example). Leaf elements in this model indicate base multimedia objects (audio, image, text, etc.). Additional attributes can be assigned to nodes in the hierarchy for conventional database management system (DBMS) access, or used in a complementary fashion (e.g., the Virtual Video Browser (VVB) [21]). Timing information is also captured with node attributes, allowing the generation of object playout times during scheduling.

2.2.2 Physical Data Structures

The TIB modeling scheme can represent an arbitrary grain of interval (e.g., individual video frames or entire movies). However, the practicality of representing each frame by its temporal parameters is limited for audio and video data that are quite homogeneous. Therefore, we group logically related frames into sets that are not normally decomposed (e.g., scenes). This aggregation corresponds to Fig. 7, and yields a blocking of sequences of continuous media. Within a block, data are assumed to have a homogeneous period of playout (Fig. 8).

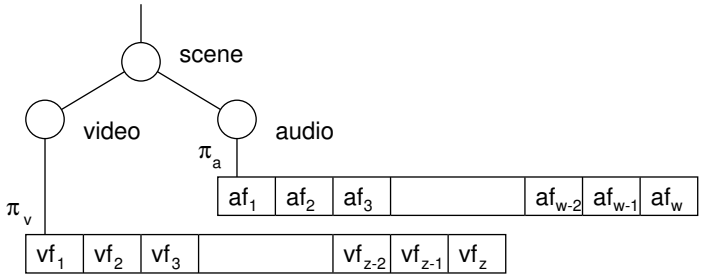

Figure 7: Blocking for Continuous Media

Once decomposed into blocks, the blocks must be mapped to the physical storage or communication medium in a manner suitable for TAC functionality. Fig. 8 shows the timing relationships within and between two streams of audio and video that must be preserved during playout. Ultimately, these *intermedia* and *intramedia* timing requirements must be satisfied by a playout mechanism (Section 2.3).

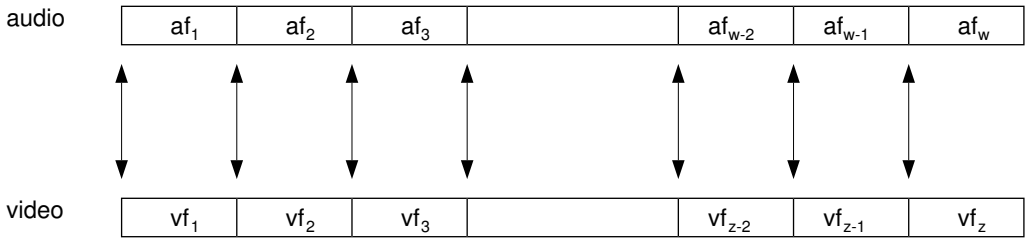

Figure 8: Timing for Audio and Video

Given a timing specification for a multimedia application, the physical system must meet these requirements. Difficulties here are due to the strict timing requirements for playout of time-dependent data. From our TIB model it is possible to derive a playout schedule appropriate for the available resources at the time of delivery. This process is achieved by

algorithmically converting the relative timing specification of the TIB model to an absolute one, and scheduling this specification on the available resources [23]. Furthermore, it is possible to derive the absolute schedule from the TIB representation in either the forward or reverse direction, or for only a fraction of the overall playout duration [24]. The resultant schedule has the form $\pi_1, \pi_2, ..., \pi_n$, where π_i are object deadlines with the property of monotonically increasing values.

2.3 Playout Scheduling

In a multimedia presentation, each medium participating in a session must be coordinated in order to have a coherent presentation at the playout device. This coordination is particularly difficult with media arriving from multiple sources because there is not a unique chain of system components involved in the delivery of data. Furthermore, because multiple processes compete for shared resources such as the communication channel, system resources that are involved in data delivery must be managed or scheduled. Changing resource requirements further complicates this problem. User interaction and the addition of new sessions contribute to these load changes. Along with dynamic inputs, a real-time multimedia scheduler must also be able to contend with limited resource availability when establishing sessions.

The playout timing for a complex multimedia object can be defined by a set of temporal relationships or as a playout schedule (a set of deadlines). Once this schedule is defined, a real-time *scheduler* must orchestrate the various resources for the desired session. The job of the real-time scheduler is to manage the assignment resources to tasks awaiting execution within the timing constraints assigned to each task. Scheduling the use of a shared resource such as a disk or communications channel can be performed by specialized real-time networks and storage systems, or by general-purpose systems that have been adapted to this task. A real-time network or storage subsystem can provide performance guarantees for data delivery. Absolute guarantees are created through deterministic scheduling and resource allocation. Approximate guarantees are offered by using statistical methods. In a real-time system of this type, the user requests specific time-based requirements and the system can then grant a connection if adequate resources exist [4, 9, 10, 16, 18, 19].

In our framework, time-dependent multimedia data delivery is accomplished by a semi-static scheduling approach using characteristics of a session that are established *a priori* from our TIB modeling scheme. Once established, fine-tuning of sessions enables graceful service degradation during periods of anomalous resource use.

2.3.1 Static Scheduling

A statistical channel can be reserved based on the source-destination delivery path characteristics by using an exact or an approximate model. The exact model is applicable to data originating from stored sources the characteristics of which can be determined *a priori*. In this case the transmission requirements of the entire data stream can be mapped to the available channel bandwidth. For the approximate source model the source data rate can be characterized by average and maximum values (e.g., live video from a camera). In either case, changes in the channel delay and bandwidth characteristics cannot be easily accommodated without disrupting the session.

The nature of both packet-switched communications and data retrieval from rotating storage devices is inherently asynchronous. We model the retrieval of data from storage and the transmission of data across a network as asynchronous system operations that must be managed in order to support the real-time data delivery required for multimedia presentation [4]. Flow-control (window) protocols can provide the feedback necessary to prevent buffer overflow in bulk data transfers. For real-time data streams, rate-based flow-control is more appropriate [9]. However, feedback approaches alone cannot provide consistent presentation quality under wide bandwidth variations due to network/storage delays and heterogeneous multimedia object sizes. For this reason, statistical reservation techniques are used.

Coarse scheduling in a multimedia session is facilitated by reservation of adequate resources. This scheduling process consists of resource reservation, connection establishment, and data transfer initiation. In relation to intermedia synchronization, a scheduler manages data transmission times with respect to the source, whereas the destination must provide buffering for delay variations and accommodation for intermedia skew.

Our framework uses a statistical scheduler based on a static reservation approach [23]. This scheduler uses parameters describing channel characteristics (Table 1) to generate a feasible delivery sequence for a multimedia object. The resultant schedule indicates the times to put objects onto the channel between the source and destination and can be used to establish the worst case buffering requirement. In the event that the source and destination clocks are unequal, periodic resynchronization can correct clock differences among involved sites and prevent rate mismatches that lead to queue underflow or overflow.

Table 1: Scheduling Parameters

$object_A$	composite multimedia object
$tree_A$	object temporal representation
$\Pi = \{\pi_i\}$	ordered sequence of object playout times (deadlines)
$\Sigma = \{\sigma_i\}$	component object sizes (bits)
D_v, D_p, D_t	queuing, propagation, and transmission delays for packet of size Sm
S_m	packet size for medium m
C	channel capacity
T_n	control time for nth block
$P(late)$	requested fraction of late packets/blocks
$\Phi = \{\phi_i\}$	ordered sequence of retrieval/transmission times (deadlines)
$\Phi^c \subseteq \Phi$	set of aperiodic deadlines from Φ

2.3.2 Dynamic Scheduling

A deficiency of the aforementioned static reservation approach is its inability to adapt to system load changes or the dynamic behavior of an interactive multimedia session. Because statistical scheduling relies on a resource commitment, changes in resource loading cannot be tolerated by the application. For this reason we are developing a hybrid statistical resource reservation approach that only performs scheduling based on a relatively short interval over the life of a session. By restricting the scheduling period we benefit from the following:

- Reduced set of deadlines to evaluate
- Rapid static scheduling performance
- Minimization of initial scheduling latency
- Adaptation to changes in resource allocation (e.g., bandwidth)
- Responsiveness to dynamic user interaction
- Close matching of required to available resources
- Tolerance to changing quality of service

This semi-static scheduling mechanism provides a balance between static and dynamic scheduling for multimedia object retrieval, transmission, and playout. We call the approach *limited a priori* (LAP) scheduling, because it is based on a static scheduling approach [23], but

it supports dynamic user input and system load changes by periodic schedule recomputation. The LAP scheduling mechanism is comprised of a static resource reservation mechanism and a dynamic, run-time executor of the LAP-produced schedule. We call these components the *LAP reservation mechanism* and the *session scheduler*, respectively. The essence of the LAP scheduling is as follows. A multimedia session is decomposed into periods of similar resource use that can be scheduled independently (Fig. 9). Each period is allocated resources and scheduled using statistical resource reservation. The resultant schedule is then executed, along with other similar schedules, by the session scheduler.

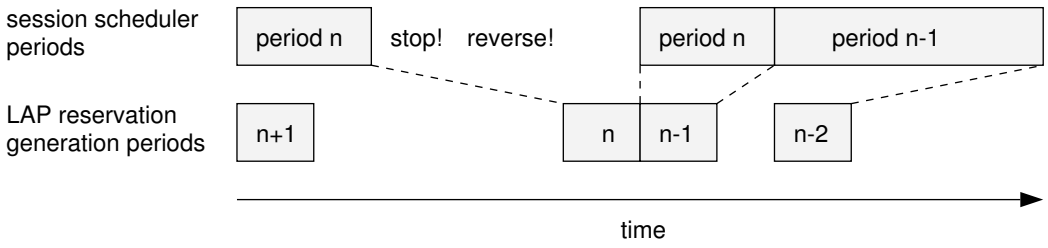

Figure 9: Schedule Creation for Multiple Periods

The LAP reservation mechanism creates a schedule for one segment, or period, of a session at a time. A new period is enacted either at the end of the current period, when a system load change is detected, or when a user initiates a TAC operation. In Fig. 9, period n is recalculated and enacted due to a user request for a reverse playout, and schedule $n-1$ is calculated so that it can be enacted at the end of period n. The length of a period L is chosen to be long enough to take advantage of bandwidth averaging, but short enough not to waste processor resources by calculating large portions of a schedule that go unused. The session scheduler is responsible for evaluating the schedule generated by the LAP reservation mechanism, responding to dynamic user input, and managing the execution of other non-real-time tasks.

For session establishment, the LAP reservation mechanism can be applied based on the characteristics of the requested session and the current system resources. The result is the generation of a set of deadlines which can be used by a retrieval/transmission process as local timing information. Included in this initialization phase are playout time identification; data size characterization; bandwidth, delay, and buffer reservation; computation of retrieval/transmission schedules; and initiation of data transfer (Table 1). The retrieval/transmission process is perceived to be independent of the monitor and playout/receive processes to support both the local and distributed data scenarios.

2.4 Fine-Tuning of Playout Timing

In the course of a multimedia application's execution, the establishment of a session can be requested, requiring the activation of the delivery and playout subsystem. Typical scenarios for this include establishing a video conference or selecting a multimedia motion picture for presentation. Once selected, the session timing requirements can be interpreted for connection establishment and maintenance. By using a statistical reservation service, the destination can be configured with a buffer of sufficient length to accommodate measured or guaranteed channel delay variations. However, changes in the channel delay distribution or violations in the bandwidth guarantee can cause buffer overflow or underflow, resulting in anomalous playout behavior.

In this section we characterize intermedia synchronization at playout time and introduce a fine-tuning mechanism to accommodate errors introduced by the LAP reservation mechanism and the session scheduler. The fine-tuning mechanism monitors and controls levels of queued periodic stream data and intermedia synchronization, and provides a graceful degradation of playout quality during periods of deviant network or storage subsystem behavior.

2.4.1 Characterization of Playout-Time Synchronization

Synchronization is defined as the occurrence of multiple events at the same instant in time. Intermedia synchronization describes a similar timing constraint among a set of multimedia streams. Timing parameters can characterize intermedia and real-time synchronization for the delivery of periodic (e.g., audio and video) and aperiodic data (e.g., text and still images). Parameters applicable to aperiodic data are *maximum delay, minimum delay,* and *average delay* as measured with respect to real time or with respect to other aperiodic data. For periodic data, maximum, minimum, and average delay are also applicable to individual data elements, but in addition, *instantaneous* delay variation or *jitter* is important for characterizing streams. *Skew*, related to jitter, describes an average jitter over an interval.

For periodic data such as audio and video, data can be lost resulting in *dropouts* or gaps in playout. Such losses cause the stream of frames to advance in time or cause a stream *lead*. Similarly, if data frames are duplicated, they cause a stream to *lag* (Fig. 10). By dropping or duplicating frames it is possible to control the rate of playback. Initiation of frame dropouts is permissible for audio and video which have substantial short-term temporal data redundancy. However, this is not true of text and graphics as their tolerances to delay are greater.

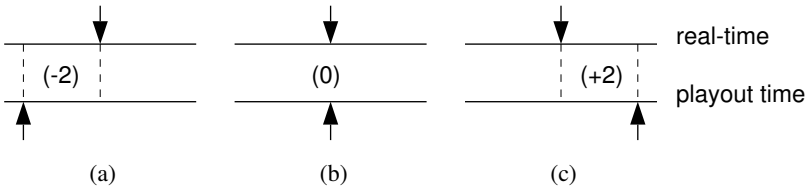

Figure 10: Skew: (a) Lagging, (b) None, (c) Leading

Synchronization is usually defined as absolute, occurring at an instant in time. To provide a tolerance to timing variations, we adopt the following definition that extends a synchronization instant to an interval (similar to Gibbs' definition [13] and Ravindran's divergence vector [29]).

Definition 1 *A composite object with actual playout times* $P = \{\rho_i\}$ *is synchronized with playout reference times* $\Pi_i = \{\pi_i\}$ *iff* $\forall i, |\rho_i - \pi_i| \leq \theta_i$ *where* $\Theta = \{\theta_i\}$ *are the synchronization tolerances between each element and the reference.*

Skew can be measured with respect to real-time as an offset to a mutual presentation start time between the source and destination, or can be measured with respect to another stream. Because many streams are possible, we characterize both intermedia and real-time reference skew for k streams using a matrix representation as $skew = sk_{p,q}$, where $sk_{p,q}$ describes the skew from stream p to stream q (q to p is negative) and the $k+1$th element corresponds to a real-time reference. We also define an interstream tolerance $\Theta = \theta_{p,q}$ and target skew matrix $TSK = tsk_{p,q}$, which indicate tolerances and target values between streams and can be interpreted by a skew control function.

Skew best measures intermedia synchronization for continuous media. For characterization of discrete events associated with timed playout of text, graphics, and still images, we can apply real-time scheduling terminology as already mentioned (e.g., maximum and minimum delay). However, it is advantageous to decompose segments of continuous media into *blocks* to permit efficient storage and manipulation, as described in Section 2.2.2. With this decomposition, blocks associate a single start deadline with a sequence of periodic media frames (Fig. 7). The playout timing of individual media elements (e.g., video frames) can then be evaluated at presentations time with respect to skew and jitter.

2.4.2 Control Paradigms

We have investigated two related control paradigms for fine-tuning of intermedia skew [25]. The first is intended to maintain intermedia synchronization between streams, or between a stream and a real-time reference, when we are operating at nominal queue levels (Fig. 11). The second is for providing a graceful degradation when queue underflow or overflow is pending and attempts at intermedia synchronization are abandoned. For intermedia synchronization, we are investigating three policies with respect to intermedia synchronization: (1) minimization of real-time skew, (2) minimization of inter-stream skew, and (3) minimization of session aggregate inter-stream skew. The first policy targets synchronization of playout with a constant end-to-end delay established during connection set-up. This policy is appropriate for streams such as high-quality digital audio. The second policy places priority on maintaining synchronization between streams rather than between a stream and a real-time reference, but could be used in conjunction with the first policy. This scenario is suitable when relative timing between media is more significant than cumulative skew with respect to real-time. The third policy is to minimize skew over a set of streams within a multimedia session. For queue-level control, our goal is to provide graceful degradation prior to queue underflow or overflow. Control takes effect upon reaching either low or high thresholds and results in modification of the playout rate via frame drop or duplication.

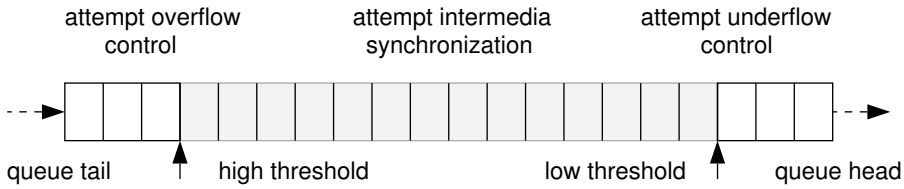

Figure 11: Three Queue States

To implement intermedia synchronization we use a control mechanism that monitors the queue level and playout skew at playout time. We provide control by changing the playout rate through dropping and duplicating frames. The stability of the system is provided by the determination of appropriate time constants for queue-level and skew measurement. The control variables *pass* and *slip* correspond to played and dropped/duplicated frames. When dropping frames, *pass* indicates the total number of frames evaluated, and *slip*, a positive integer, corresponds to the number to drop from *pass* passing frames. When duplicating frames, *pass* has the same interpretation, however, *slip* indicates the number of duplications of the last frame in the sequence. This formulation leads to a fast evaluation at playout

time.

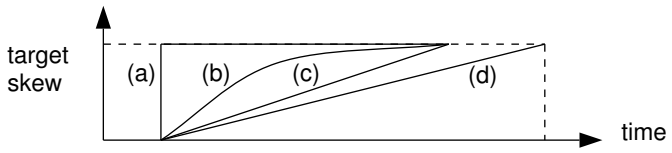

Figure 12: Correction Alternatives. (a) No Control, (b) Non-Linear, (c) Constant-Rate, (d) Constant-Time

To set the control variables *pass* and *slip*, we define control functions that are dependent on the tolerable rate of drop and duplication for each medium. These functions can be instantaneous, introducing a large discontinuity in playout, or can be more gradual (Fig. 12). Our control algorithm supports any type of selective drop or duplication function. We currently use a constant rate-based correction function, but other functions can be supported as well. The control algorithm is outlined below.

1. for each stream k do
   (a) if $qlevel_k < qll_k$ then {low queue level}
      - $(pass, slip) := under(qlevel_k)$ {initiate stream lag}
   (b) elsif $qlevel_k > qlh_k$ then {high queue level}
      - $(pass, slip) := over(qlevel_k)$ {initiate stream lead}
   (c) else {nominal queue level}
      - $(pass, slip) := sync(qlevel_k, skew(k, l))$ {synchronize stream k to l}
2. {update queue level, skew, and lost frame statistics}

On each iteration, this algorithm invokes one of the control functions under(), over(), or sync() depending on the queue level. If it is high or low, the algorithm provides service degradation in the form of drops or duplications until service is restored. If the level is nominal, then intermedia synchronization control is applied. In all cases, the values of *pass* and *slip* are manipulated to control the playout rate and to effectively control skew.

3 Conclusion

In this paper we have described our framework for managing time-dependent data in a multimedia information system. The framework comprises components designed to support multimedia presentation authoring, storage, interaction, and playback.

Individual components of the framework that have been exercised are currently being combined in the construction of a general purpose, distributed, multimedia information system in the Multimedia Communications Laboratory at Boston University. As part of this work we have implemented an interactive motion-picture database application called the Virtual Video Browser System [21]. This system is representative of multimedia applications and illustrates the use of temporal models and database structures to support TAC functionality.

References

[1] Anderson, D.P. and G. Homsy, "A Continuous Media I/O Server and Its Synchronization Mechanism," *Computer*, Vol 24, No. 10, October 1991, pp. 51-57.

[2] Bulterman, D.C.A., G. van Rossum, and R. van Liere, "A Structure for Transportable, Dynamic Multimedia Documents," *Proc. Summer 1991 Usenix Conf.*, Nashville, Tennessee, June 1991, pp. 137-155.

[3] Bulterman, D.C.A. and R. van Liere, "Multimedia Synchronization and UNIX," *Proc. 2nd Intl. Workshop on Network and Operating Support for Digital Audio and Video*, Heidelberg, Germany, November 1991, pp. 108-119.

[4] Chen, H.J. and T.D.C. Little, "Physical Storage Organizations for Time-Dependent Multimedia Data," to appear at the 4th Intl. Conf. on Foundations of Data Organization and Algorithms (FODO'93), Evanston, IL, October 1993.

[5] Christodoulakis, S. and C. Faloutsos, "Design and Performance Considerations for an Optical Disk-based, Multimedia Object Server," *Computer*, Vol. 19, December 1986, pp. 45-56.

[6] Committee Draft International Standard, "Information Technology Hypermedia/Time-based Structuring Language (HyTime)," ISO/IEC CD 10743, April 1, 1991.

[7] Dannenberg, R.B., "Remote Access to Interactive Media," *Proc. SPIE Symposium OE/FIBERS'92, (Enabling Technologies for Multi-Media, Multi-Service Networks)*, Boston, MA, September 1992, pp. 230-237.

[8] Dannenberg, R.B., D. McAvinney, and P. Rubine, "Arctic: A Functional Approach to Real-Time Control," *Computer Music Journal*, 10(4) (Winter 1986), pp. 67-78.

[9] Ferrari, D. and D.C. Verma, "A Scheme for Real-Time Channel Establishment in Wide-Area Networks," *IEEE J. Selected Areas in Communications*, Vol. 8, No. 3, April 1990, pp. 368-379.

[10] Field, B. and T. Znati, "α-Channel, A Network Level Abstraction to Support Real-Time Communication," *Proc. 2nd Intl. Workshop on Network and Operating Support for Digital Audio and Video*, Heidelberg, Germany, November 1991, pp. 148-159.

[11] Fujikawa, K., S. Shimojo, T. Matsuura, S. Nishio, and H. Miyahara, "Multimedia Presentation System "Harmony" with Temporal and Active Media," *Proc. Summer 1991 Usenix Conf.*, Nashville, Tennessee, June 1991, pp. 75-93.

[12] Gemmell, J. and S. Christodoulakis, "Principles of Delay-Sensitive Multimedia Data Storage and Retrieval," *ACM Trans. on Information Systems*, Vol. 10, No. 1, January 1992, pp. 51-90.

[13] Gibbs, S., L. Dami, and D. Tsichritzis, "An Object-Oriented Framework for Multimedia Composition and Synchronisation," in *Object Composition*, Tech. Rept., University of Geneva, June 1991, pp. 133-143.

[14] Herrtwich, R.G., "Time Capsules: An Abstraction for Access to Continuous-Media Data," *Proc. 11th Real-Time Systems Symp.*, Lake Buena Vista, FL, December 1990, pp. 11-20.

[15] Herrtwich, R.G. and L. Delgrossi, "ODA-Based Data Modeling in Multimedia Systems," *International Computer Science Institute Tech. Rept.* TR-90-043, August 1990.

[16] Jeffay, K., D.L. Stone, T. Talley, and F.D. Smith, "Adaptive Best-Effort Delivery of Digital Audio and Video Across Packet-Switched Networks," *Proc. Third International Workshop on Network and Operating System Support For Digital Audio and Video*, San Diego, CA, November 1992, pp. 1-12.

[17] Kipp, N., V. Newcomb, and S. Newcomb, "Hytime Review," TechnoTeacher Inc., 1990.

[18] Lazar, A.A., A. Temple, and R. Gidron, "An Architecture for Integrated Networks that Guarantees Quality of Service," *International Journal of Digital and Analog Cabled Systems*, Vol. 3, No. 2, 1990, pp. 229-238.

[19] Lim, C.C., L. Yao, and W. Zhao, "Transmitting Time-Dependent Multimedia Data in FDDI Networks," *Proc. SPIE Symposium OE/FIBERS'92*, Boston, MA, September 1992.

[20] Lippman, A., "Movie-Maps: An Application of the Optical Videodisc to Computer Graphics," *Computer Graphics*, July 1980, (SIGGRAPH '80 Conf. Proc., Seattle, WA, July 1980), pp. 32-42.

[21] Little, T.D.C., G. Ahanger, R.J. Folz, J.F. Gibbon, F.W. Reeve, D.H. Schelleng, and D. Venkatesh, "A Digital Video-on-Demand Service Supporting Content-Based Queries," *Proc. ACM Multimedia'93*, Anaheim CA, August 1993, pp. 427-436.

[22] Little T.D.C. and A. Ghafoor, "Synchronization and Storage Models for Multimedia Objects," *IEEE J. on Selected Areas in Comm.*, Vol. 8, No. 3, April 1990, pp. 413-427.

[23] Little, T.D.C. and A. Ghafoor,"Scheduling of Bandwidth-Constrained Multimedia Traffic," *Computer Communications*, Vol. 15, No. 6, pp. 381-387, July/August 1992.

[24] Little, T.D.C., and A. Ghafoor, "Interval-Based Conceptual Models for Time-Dependent Multimedia Data," to appear in *IEEE Transactions on Knowledge and Data Engineering*, August 1993.

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

[26] Nakajima, J., M. Yazaki, and H. Matsumoto, "Multimedia/Realtime Extensions for the Mach Operating System," *Proc. Summer 1991 Usenix Conf.*, Nashville, Tennessee, June 1991, pp. 183-198.

[27] Nicolaou, C. "An Architecture for Real-Time Multimedia Communication Systems," *IEEE J. Selected Areas in Comm.*, Vol. 8, No. 3, April 1990, pp. 391-400.

[28] Rangan, P.V. and H.M. Vin, "Designing File Systems for Digital Video and Audio," *Proc. 13th Symp. on Operating Systems Principles (SOSP'91), Operating Systems Review*, Vol 25, No. 5, October 1991, pp. 81-94.

[29] Ravindran, K., "Real-Time Synchronization of Multimedia Data Streams in High Speed Networks," *Proc. 1992 Workshop on Multimedia Information Systems*, Tempe, Arizona, February, 1992, pp. 164-188.

[30] Sasnett, R.M., "Reconfigurable Video," *M.S. Thesis*, Department of Architecture, MIT, 1986.

[31] Steinmetz, R., "Synchronization Properties in Multimedia Systems," *IEEE J. Selected Areas in Comm.*, Vol. 8, No. 3, April 1990, pp. 401-412.

[32] Stotts, P.D., Furuta, R., "Petri-Net-Based Hypertext: Document Structure with Browsing Semantics," *ACM Trans. on Office Automation Systems*, Vol. 7, No. 1, Jan. 1989, pp. 3-29.

[33] Szabo, B.I. and G.K. Wallace, "Design Considerations for JPEG Video and Synchronized Audio in a Unix Workstation," *Proc. Summer 1991 Usenix Conf.*, Nashville, Tennessee, June 1991, pp. 353-368.

[34] Wolf, L.C., "A Runtime Environment for Multimedia Communications," Presented at the 2nd Intl. Workshop on Network and Operating Support for Digital Audio and Video, Heidelberg, Germany, November 1991.

[35] Yu, C., W. Sun, D. Bitton, Q. Yang, R. Bruno, and J. Tullis, "Efficient Placement of Audio Data on Optical Disks for Real-Time Applications," *Comm. of the ACM*, Vol. 32, No. 7, July 1989, pp. 862-871.

[36] Zellweger, P.T., "Toward a Model for Active Multimedia Documents," in *Multimedia Interface Design*, M.M. Blattner and R.B. Dannenberg, Eds., ACM Press, New York, NY, 1992, pp. 39-52.