Scheduling of Bandwidth-Constrained Multimedia Traffic

T.D.C. Little
Department of Electrical, Computer and Systems Engineering
Boston University, Boston, MA 02215 USA
tdcl@buenga.bu.edu

A. Ghafoor
School of Electrical Engineering
Purdue University, West Lafayette, IN 47907 USA

Abstract - Multimedia applications describe unique requirements that must be met by computer network and operating system components. In particular, the time-dependencies of multimedia data require mechanisms to ensure timely and predictable delivery of data from their sources to destinations. For single medium applications which have relatively constant bandwidth utilization, connections from source to destination can be tailored to moderate ranges of data rates. On the other hand, due to the large variation in multimedia object sizes and concurrency in object presentation, multimedia applications can require a correspondingly large variation in required bandwidth over the life of a connection. Data of these types may not arrive in time to meet the intended playout schedule when the capacity of the channel is exceeded. In this paper we present an approach to remedying this situation by effectively smoothing the bandwidth requirement over time via a scheduling mechanism.

1 Introduction

One of the unique requirements of a multimedia information system (MMIS) is the capability to present time-dependent data to the user. Time dependencies of multimedia data can be implied during creation, e.g., sequences of live video images. Data can also have time dependencies formulated explicitly, e.g., a pre-orchestrated slide presentation [1]. In either case, these time dependencies must be characterized and supported by the system. A MMIS must also be able to overcome any system delays caused by storage, communication, and computational latencies in the procurement of data for the user. These latencies are usually random in nature, being caused by shared access to common system resources. In a distributed multimedia information system (DMIS), multiple data sources including databases, video cameras, and telephones can be connected to user workstations via high-speed packet-switched networks. System latencies in a DMIS are problematic since several streams originating from independent sources can require synchronization to each other in spite of the asynchronous nature of the network.

To support the presentation of time-dependent data, scheduling disciplines associated with real-time operating systems are necessary. However, the real-time requirements can be relaxed since data delivery can tolerate some occasional lateness, i.e., catastrophic results do not occur when data are not delivered on time. For multimedia data, real-time deadlines consist of the playout times for individual data elements that can be scheduled based on a specified probability being missed. The design of a system to support time-dependent media must account for latencies in each system component used in the de-

livery of data, from its source to its destination. Therefore, specific scheduling is required for storage devices, the CPU, and communications resources.

Most recent work supporting time-dependent data is in the areas of live data communications [2-10], data storage systems [1, 11-17], and general distributed systems [1, 8, 18-22]. The work in data communications has usually been applied to live, periodic data sources such as packet audio and video [2-10] and for applications that do not exceed the capacity of the communication channel. These applications typically use a point-to-point configuration since there is seldom any need to deliver synchronous data streams from multiple independent live sources. Packet delay variations at the receiver are effectively eliminated by buffering and the introduction of a constant time offset resulting in a reshaping of the distribution of arriving packets to reduce delay variance.

For stored-data applications, time-dependencies must also be managed [1, 11-17]. Unlike live data, which are typically periodic during acquisition and presentation, stored data can require aperiodic playout and can be retrieved from storage at arbitrary times prior to presentation to the user. Audio and video data types require very large amounts of storage space and will exist primarily in secondary storage. When data originate from storage, the system has more flexibility in scheduling the times at which data are communicated to the application. Since data are not generated in real-time, they can be retrieved in bulk, well ahead of their playout deadlines. This is contrasted with live sources, e.g., a video camera, that generate data at the same rate as required for consumption. Unlike live sources, stored data applications in a DMIS can require synchronization for data originating from independent sources and simple data sequencing cannot provide intermedia synchronization.

Our work is intended for stored-data applications which, due to the heterogeneity and concurrency of multimedia data, the channel capacity can be exceeded for some intervals. We propose the use of *a priori* knowledge of data traffic characteristics to facilitate scheduling, when available, rather than providing on-line scheduling of dynamic input data based on its statistical nature. In essence, the dynamic bandwidth requirements of a multimedia object are fit into a finite channel capacity rather than the available bandwidth dictating the feasibility of the application, i.e., the inherent burstiness of the object is smoothed via our approach. The result is that delays are traded-off for the satisfaction of a playout schedule, even when the capacity of the channel is exceeded by the application.

In the remainder of this paper we describe an approach to the scheduling of time-dependent multimedia data retrieval under limited bandwidth conditions. In Section 2, we discuss the properties of time-dependent multimedia data. In Section 3, we describe our approach to scheduling time-dependent traffic under a bandwidth constraint, and provide several examples. Section 4 describes practical considerations for applying the derived schedules. Section 5 concludes the paper.

2 Properties of Time-Dependent Data

For management of time-dependent data in a computer system we are confronted with sequences of data objects which require periodic and aperiodic computation and communication that is oriented towards presentation of information to the user. In this section we describe several approaches to characterizing these time-dependencies through temporal modeling. These models can then facilitate real-time scheduling of data presentation when system latencies are present (Section 3).

2.1 Temporal Models

A common representation for the temporal component of time-dependent data is based on temporal intervals [12, 23, 24] in what is called temporal-interval-based (TIB) modeling. This technique associates a time interval to the playout duration of each time-dependent data object. For example, the playout of a sequence of video frames can be represented by using temporal intervals as shown in Fig. 1. A binary relationship called a *temporal relationship* (*TR*) can be identified between pairs of intervals. A temporal relationship can indicate whether two intervals overlap, abut, etc. [24]. With such a TIB modeling scheme, complex timeline representations of multimedia object presentation can be represented by application of the temporal relations to pairs of temporal intervals. Each interval can also be represented by its *start time* and *end time* via instant-based modeling. Therefore, for a sequence of data objects, we can describe their time dependencies as either a sequence of related intervals or as a sequence of start and end times. In the latter, the start times can be interpreted as *deadlines* as required for real-time scheduling. Fig. 2 shows the symbols that we associate with the intervals and deadlines in a time-dependent data specification. For each interval i we indicate its start time π_i and duration τ_i. The relative positioning of multiple intervals represents their time dependencies τ_δ^i. Their overall duration is defined by τ_{TR}.

A disadvantage with the instant-based approach is that modifications, including deletions or insertions, to a sequence of time-dependent data require changing subsequent time instances. For a TIB representation, the deadlines are implied by the precedence relationships described by the temporal relations. However, time instances in the form of deadlines are more suitable for the real-time scheduling during data playout [25]. To satisfy both needs we need the ability to derive the deadlines from the TIB representation.

2.2 *n*-ary Temporal Relationships

Binary temporal relations are sufficient for describing any simple or complex time-dependent data. However, we can generalize the binary temporal relations and ultimately simplify the data structures necessary for maintaining temporal semantics. Furthermore, by restricting the temporal relationships to have certain characteristics, we can simplify the process of scheduling data retrieval of time-dependent data. We therefore propose a new kind of homogeneous temporal relation for describing time-dependent data. The new temporal relation on n objects, or intervals, is defined as follows:

Definition 1: Let P be an ordered set n of temporal intervals such that $P = \{P_1, P_2, \ldots P_n\}$. A temporal relation, *TR*, is called an n-ary temporal relation, denoted TR^n, if and only if

$$P_i \; TR \; P_{i+1} \;\; \forall i \; (1 \le i < n).$$

Like binary temporal relations, there are thirteen possible n-ary temporal relations which reduce to seven cases after eliminating their inverses [12]. When $n = 2$, the n-ary temporal relations simply become the binary ones, i.e., when $n = 2$, $P_1 \; TR \; P_2$. If we restrict our discussion to n-ary relations with the property that

$$\pi_i \le \pi_j, \;\; \forall i, j, \;\; i < j, \; (1 \le i, j \le n),$$

then the deadlines of each object in a given sequence will be monotonically increasing

[26]. This property is satisfied by the temporal relations of *before, meets, overlaps, during*$^{-1}$, *starts, finishes*$^{-1}$, and *equals*. By using this subset of the thirteen temporal relations we do not restrict our ability to model time-dependencies [26].

In the process of presenting time-dependent multimedia data, we can use precedence relationship between related intervals [12], or we can use a deadline-driven approach. Since the *n*-ary temporal representation uses precedence relationships among intervals, to apply a deadline-driven approach we must be able to generate the exact playout deadline of each element based on the intervals and their relationship. This task is necessary for real-time scheduling of object retrieval in the presence of non-negligible delays [25]. The following theorem describes the relative playout time (deadline) for any object or start point of a temporal interval.

Theorem 1: The relative deadline π_k for interval k for any n-ary temporal relation, is determined by

$$\pi_k = 0, \text{ for } k = 1$$

$$\pi_k = \Sigma_{i=1}^{k-1} \tau_\delta^{\,i}, \text{ for } 1 < k \leq n$$

Proof: Since timing is relative to the start of the set of intervals, we let $\pi_1 = 0$. $\pi_2 = \pi_1 + \tau_\delta^{\,1}$ since $\tau_\delta^{\,1} = \pi_2 - \pi_1$, by definition of delay for the binary case. Suppose that $\pi_m = \Sigma_{i=1}^{m-1} \tau_\delta^{\,i}$, for some m. For intervals P_m and P_{m+1}, P_m *TR* P_{m+1} by Definition 1. Therefore, we can conclude the hypothesis by applying induction on π_m. ||

Theorem 1 describes the efficient conversion of an interval-based representation to an instant-based one. In the following section we investigate the implications of real system latencies on the retrieval of time-dependent data and the satisfaction of the temporal specification.

2.3 Playout Timing with Delays

In the presence of real system latencies, several additional delay parameters are introduced as shown in Fig. 3. In order to properly synchronize some data element x with a playout time π, sufficient time must be allowed to overcome the latency λ caused by data generation, packet assembly, transmission, etc., otherwise, the deadline π will be missed. If we choose a time called the *control time* T such that $T \geq \lambda$, then scheduling the retrieval at T time units prior to π guarantees successful playout timing. The *retrieval time*, or packet production time ϕ is defined as $\phi = \pi - T$. The case of synchronizing one event is equivalent to the problem of meeting a single real-time deadline, a subset of the real-time scheduling problem. For streams of data, the multiple playout times and latencies are represented by the sets $\Pi = \{\pi_i\}$ and $\Phi = \{\phi_i\}$.

Fig. 4 illustrates the effect of introducing a control time to reduce the delay variation on the elements of a sequence of data called a *stream*. Here, elements of stream are generated at a source and experience a random delay with a distribution described by $p(t)$ before reaching their destination. They then require synchronization based on a playout schedule with deadlines π_i and playout durations τ_i. A suitable control time can be found for a stream of data and target percentage of missed deadlines based on such a delay characteristic [2-5]. Variation in both λ_i, the latencies of individual stream elements, and τ_i, their playout durations, can result in short term average or instantaneous need for buffering which is accommodated by T. For live sources, if ϕ_i are the packet production times, then $\phi_i = \pi_i - T$, $\forall i$, where π_i are the playout times. The playout sequence is merely

shifted in time by T from the generation times. Furthermore, playout durations are typically uniform ($\tau_i = \tau_j, \forall i, j$), and delay variation governs the buffering requirement. In the general case, we are presented with arbitrary streams characterized by π_i, λ_i, and τ_i and would like to determine the amount of buffering necessary to maintain playout timing with a given probability of failure.

One of the assumptions in the earlier analyses is that the generation of packets is at a rate equal to the consumption rate as is typical for live sources. When we consider a database as the source of a packetized stream instead of a live source (e.g., a camera), it is permissible for the playout schedule to specify retrieval of data exceeding the available channel capacity, and the earlier scheduling policies are deficient. In essence, data storage sources give us more freedom in controlling the packet production times. This is particularly important when we require concurrency in the playout of multiple media.

3 Scheduling with a Bandwidth Constraint

Since it is assumed that the channel capacity can be exceeded, we seek a method for smoothing the overloaded capacity during the life of a connection interval yet still maintaining playout timing. In this section we present our proposed scheduling approach and illustrate it with several examples. The approach generates a schedule that utilizes the full capacity of the channel by providing buffering at the destination based on *a priori* knowledge of the playout schedule Π.

3.1 Data Retrieval Timing

To develop an approach to scheduling data retrieval we must first characterize the properties of the multimedia objects and the communications channel. The total end-to-end delay for a packet can be decomposed into three components: a constant delay D_p corresponding to propagation delay and other constant overheads, a constant delay D_t proportional to the packet size, and a variable delay D_v which is a function of the end-to-end network traffic. D_t is determined from the channel capacity C as $D_t = S_m/C$, where S_m is the packet size for medium m. Therefore, the end-to-end delay for a single packet can be described as

$$D_e = D_p + D_t + D_v.$$

Since multimedia data objects can be very large, they can consist of many packets. If $|x|$ describes the size of some object x in bits, then the number of packets r constituting x is determined by $r = \text{ceiling}(|x|/S_m)$. The end-to-end delay for an object x is then

$$D_e = D_p + rD_t + \sum_{j=1}^{r} D_{v_j}.$$

Like the non-bandwidth constrained approaches, a control time can be identified for either single packets or for complete objects given a selected probability of receiving a late packet (object) called *P(fail)*. Control time T_i is defined as the skew between putting a packet (object) onto the channel and playing it out ($T_i = \pi_i - \phi_i$). Given *P(fail)*, and the cumulative distribution function F of interarrival delays on the channel, we can readily find T_i for either packets or objects as shown below.

$$T_i = D_p + S_m/C + F^{-1}(1 - P(fail))$$
$$T_i = D_p + r_i S_m/C + F_r^{-1}(1 - P(fail))$$

In these equations, $F^{-1}(p)$ determines the delay d for $p = F(d) = P(D_v \leq d)$ and $F_r^{-1}(p)$ determines the delay d for $p = F_r(d) = P(\sum_{j=1}^{r} D_{v_j} \leq d)$. Since $F_r(d)$ requires a convolution on r random variables, it is computationally impractical for an arbitrary distribution on D_v when r is large. However, a Gaussian approximation for the distribution of a sum of r independent random variables is reasonable for large r, and can be realized given the mean μ, and variance σ^2, of the interarrival distribution.

3.2 Delay and Capacity Constraints

In Section 3.1, only single packets and objects are considered. When multiple objects are retrieved, it is possible to exceed the capacity of the channel, and therefore we must consider the timing interaction of the objects. Transmission of objects is either back-to-back or introduces slack time, depending on their size and time of playout. We define an optimal schedule for a set of objects to be one which minimizes their control times. Two constraints determine the minimum control time for a set of objects. These are the minimum end-to-end delay (MD constraint) per object, and the finite capacity (FC constraint) of the channel. The MD constraint simply states that an object cannot be played-out before arrival. This constraint must always be met. The FC constraint describes the relation between a retrieval time and its successor when the channel is busy and accounts for the interarrival time due to transit time and variable delays. It also represents the minimum retrieval time between successive objects. These constraints are summarized mathematically below.

$$\pi_i \geq \phi_i + T_i \qquad \text{MD constraint}$$
$$\phi_{i-1} \leq \phi_i - T_{i-1} + D_p \qquad \text{FC constraint}$$

The following result combines the MD and FC constraints and describes the characteristics of an optimal schedule (see Fig. 5).

Theorem 2: An optimal schedule for any object i has the following characteristics:

$$\forall i,\ \phi_i \geq \pi_{i-1} - D_p \Rightarrow \phi_{i-1} = \pi_{i-1} - T_{i-1}, \text{ and } \phi_i < \pi_{i-1} - D_p \Rightarrow \phi_{i-1} = \phi_i - T_{i-1} + D_p$$

Proof: Any object in the set can find the channel idle or with a backlog of items. If it is slack, the equality of the MD constraint is optimal, by definition. Similarly, when the channel is busy, the equality of the FC constraint is optimal. Clearly the channel is slack when ϕ_i occurs after π_{i-1}, but it can also be slack when $\phi_i \geq \pi_{i-1} - D_p$ due to the "pipeline" delays. Therefore, the state of the channel, idle or slack, can be identified. ||

We now show how to construct such a schedule. Given the characteristics of the channel and of a composite multimedia object (D_v, D_p, D_t, C, S_m, π_i, $|x_i|$), a schedule can be constructed. We begin by establishing an optimal retrieval time for the final, mth object, i.e., $\phi_m = \pi_m - T_m$. The remainder of the schedule can be determined by iterative application of the MD and FC constraints for adjacent objects. The resultant schedule indicates the times at which to put objects onto the channel between source and destination, or it can be used to establish the worst-case buffering requirement. This approach is embodied in the following scheduling algorithm shown below.

$\phi_m = \pi_m - T_m$
for $i = 0 : m - 2$
 if $\phi_{m-i} < \pi_{m-i-1} - D_p$
  $\phi_{m-i-1} = \phi_{m-i} - T_{i-1} + D_p$
 else
  $\phi_{m-i-1} = \pi_{m-i-1} - T_{m-i-1}$
 end
end

It is also possible to determine the number of buffers required when using the playout schedule Φ, again noting that it is necessary to obtain an element prior to its use. For each fetch time ϕ_i, the number of buffers K_i required is equal to the size of the objects with elapsed playout deadlines between ϕ_i and π_{i-1} [25].

From this methodology we can determine the buffer use profile, ϕ_i versus K, and the maximum delay incurred by buffering, which can be used for allocation of storage resources. We now show several examples of the application of the scheduling approach.

3.3 Examples

Fig. 6 illustrates the time-dependencies of a sequence of full-motion video images represented by the OCPN [12]. The video is assumed to be compressed and has a nominal playout rate of 30 frames/sec. ($|x_i| = 1 \times 10^6$ bits and $\tau_i = 1/30$ sec.). The following conditions are assumed for the communications channel: $C = 1.5$ Mbit/sec., $S_m = 8192$ bits, $D_v = 50$ μsec., and $D_p = 100$ μsec. The results of the scheduling algorithm are:

$$\Pi = \{0.0, 0.033, 0.067, 0.10, 0.13, 0.17\} \text{ sec.}$$
$$\Phi = \{-0.028, 0.0057, 0.039, 0.072, 0.11, 0.14\} \text{ sec.}$$
$$K = \{0, 0, 0, 0, 0, 0\} \text{ bits}$$

These results indicate that there is ample time to transmit the specified video traffic without buffering (K buffers) beyond the object in transit, i.e., the size and playout timing is not affected by the channel capacity constraint.

Fig. 7 shows a more complex example which exceeds channel capacity. In this case, audio (64 Kbits/sec.), video and images (1024 x 1024 pixels x 8 bits/pixel) are coordinated in a multimedia presentation. Video and audio data elements are retrieved in 5 and 10 sec. segments, respectively. The results of the scheduling algorithm applied to the playout sequence are:

$$\Pi = \{0, 0, 0, 5, 10, 10, 15, 20, 20, 20, 25, 30, 30, 35\} \text{ sec.}$$
$$\Phi = \{-9.5, -6.1, -5.7, 0.0, 0.0, 3.4, 3.8, 7.2, 10.5, 11.0, 16.7, 21.6, 26.2, 26.6\} \text{ sec.}$$
$$K = \{0, 50, 100, 184, 50, 100, 184, 184, 100, 184, 184, 50, 50, 100\} \times 10^5 \text{ bits}$$

In this case, the scheduling algorithm produces a retrieval schedule based on scheduling adjacent objects in the channel. The results indicate that an initial delay of 9.5 sec. is required prior to beginning playout at the receiver.

We now consider practical aspects of the use of the scheduling approach including determination of network delays and application of the derived schedule.

4 Implementation Issues

In this section we show how the derived schedule Φ can be used in two ways. First, the sender can use it to determine the times at which to put objects onto the channel. The receiver then buffers the incoming objects until their deadlines occur. In this case the source must know Φ, the destination must know Π, and the overall control time is T_1 (the computed delay of the first element in the sequence). This scheme can be facilitated by appending π_i onto objects as they are transmitted in the form of time stamps. The second way to use Φ is to determine the worst-case skew between any two objects as $T_W = \max(\{\pi_i - \phi_i\})$, and then to provide T_W amount of delay for every object. Transmission scheduling can then rely on a fixed skew T_W and the Π schedule, that is, it can transmit all items with π_i in the interval $(t \leq t + T_W)$.

The first method minimizes both buffer utilization and delay since the schedule is derived based on these criteria. Furthermore, required buffer allocation need not be con stant but can follow the buffer use profile. The second method provides unnecessary buffering for objects, well ahead of their deadlines, and requires constant buffer allocation. However, it provides a simpler mechanism for implementation. Consider a long sequence of objects of identical size and with regular (periodic) playout intervals. For this sequence, $T_1 = T_W$, and $\phi_i = \pi_i - T_W$, assuming the channel capacity C is not exceeded. In this case, the minimum buffering is also provided by the worst-case delay. Such a se quence describes constant bit rate (CBR) video or audio streams which are periodic, producing fixed size frames at regular intervals. Rather than manage many computed deadlines/sec. (e.g., 30/sec. for video), a transmission mechanism more simply sequentially transmit objects based on sequence number and T_W.

When data originate from live sources, the destination has no control over packet generation times, and sufficient channel capacity must be present to preserve the real-time characteristic of the streams. In this case, the control time can be determined from the size, delay, and channel capacity requirements of a representative data object, e.g., a CBR video frame, and only provides jitter reduction at the receiver. For variable bit rate (VBR) objects, the source data stream is statistical in size and frequency of generated objects, and we apply a pessimistic worst-case characterization of the largest and most frequent VBR object to determine the control time. $\{\phi_i\}$ defines the packet production times, and playout times are $\pi_i = \phi_i + T$, as done in previous work. This service can be supported by appending timestamps, in the form of individual deadlines, π_i, to the transmitted objects.

In the general case, a playout schedule is aperiodic, consisting of some periodic and aperiodic deadlines. Typically, during class decomposition, these are isolated, and one of the two scheduling schemes above can be applied. If both exist in the same playout schedule Π (e.g., if classes are not decomposed), then the derived schedule Φ can be used as follows: choose a new control time T_E such that $(T_1 \leq T_E \leq T_W)$, and drop all deadlines ϕ_i from Φ such that $T_E \geq \pi_i - \phi_i$. The result is a culled schedule Φ reflecting the deadlines that will not be satisfied by simply buffering based on T_E. By choosing T_E to encompass a periodic data type (e.g., video), the burden of managing periodic deadlines is eliminated, yet aperiodic objects, requiring extensive channel utilization, can be dealt with using $\phi_i - T_E$, where $\phi_i - T_E > 0$.

When multiple delay channels are cascaded, e.g., channel delays plus storage device delays, we see that the end-to-end path capacity is reduced to the value of the slowest component. To facilitate scheduling of each resource, we can apply the derived schedule

Φ of the first channel as the playout schedule of the succeeding channel, while moving towards the source. In this manner, each resource merely meets the schedule stipulated by the preceding resource in the chain, with the initial schedule as Π. For this scheme, however, to meet the same *P(fail)*, the failure probability used for computing Φ on each link must be divided between each component [27].

5 Conclusion

Multimedia applications require the ability to store, communicate, and playout time-dependent data. In this paper we present an approach to satisfying timing requirements in the presence of real system delays when a limited bandwidth component such as a communications channel or storage device is present. The approach utilizes a timing specification in the form of a set of monotonically increasing playout times which are shown to be derivable from a temporal-interval-based timing specification. From the timing specification a retrieval schedule is computed based on the characteristics of the limited-capacity resource. The examples presented illustrate the utility in the proposed mechanism for applications which would normally not be viable due to timing specification violation during multimedia object retrieval.

6 References

[1] Little, T.D.C., Ghafoor, A., "Network Considerations for Distributed Multimedia Object Composition and Communication," *IEEE Network*, Vol. 4, No. 6, November 1990, pp. 32-49.

[2] Barberis, G., Pazzaglia, D., "Analysis and Optimal Design of a Packet-Voice Receiver," *IEEE Trans. on Comm.*, Vol. COM-28, No. 2, February 1980, pp. 217-227.

[3] Barberis, G., "Buffer Sizing of a Packet-Voice Receiver," *IEEE Trans. on Comm.*, Vol. COM-29, No. 2, February 1981, pp. 152-156.

[4] Montgomery, W.A., "Techniques for Packet Voice Synchronization," *IEEE J. on Selected Areas in Comm.*, Vol. SAC-1, No. 6, December 1983, pp. 1022-1028.

[5] De Prycker, M., "Functional Description and Analysis of a Video Transceiver for a Broad Site Local Wideband Communications System," *Esprit '85: Status Report of Continuing Work*, The Commission of the European Communities, Ed., North-Holland, New York, 1986, pp. 1087-1108.

[6] De Prycker, M., Ryckebusch, M., Barri, P., "Terminal Synchronization in Asynchronous Networks," *Proc. ICC '87* (IEEE Intl. Conf. on Comm. '87), Seattle, WA, June 1987, pp. 800-807.

[7] Naylor, W.E., Kleinrock, L., "Stream Traffic Communication in Packet Switched Networks: Destination Buffering Considerations," *IEEE Trans. on Comm.*, Vol. COM-30, No. 12, December 1982, pp. 2527-2524.

[8] Adams, C., Ades, S., "Voice Experiments in the UNIVERSE Project," *IEEE ICC '85 Conf. Record*, Chicago, IL, 1985, pp. 29.4.1-29.4.9.

[9] Ades, S., Want, R., Calnan, R., "Protocols for Real Time Voice Communication on a Packet Local Network," *Proc. IEEE INFOCOM '87*, San Francisco, CA, March, 1987, pp. 525-530.

[10] Ma, J., Gopal, I., "A Blind Voice Packet Synchronization Strategy," *IBM Research Rept.* RC 13893 (#62194) Watson Research Center, July 1988.

[11] Salmony, M.G., Sheperd, D., "Extending OSI to Support Synchronization Required by Multimedia Applications," *IBM ENC Tech. Rept.* No. 43.8904, April 1989.

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

[13] Postel, J., Finn, G., Katz, A., and Reynolds, J., "An Experimental Multimedia Mail System," *ACM Trans. on Office Information Systems*, Vol. 6, No. 1, January 1988, pp. 63-81.

[14] Ghafoor, A., Berra, P., and Chen, R., "A Distributed Multimedia Database System," *Proc. Workshop on the Future Trends of Distributed Computing Systems in the 1990s*, Hong Kong, September 1988, pp. 461-469.

[15] Gemmell, J., Christodoulakis, S., "Principles of Delay Sensitive Multi-Media Data Retrieval," *Proc. 1st Intl. Conf. on Multimedia Information Systems '91*, Singapore, January 1991, McGraw-Hill, Singapore, pp. 147-158.

[16] Yu, C., Sun, W., Bitton, D., Yang, Q., Bruno, R., Tullis, J., "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.

[17] Wells, J., Yang, Q., Yu, C., "Placement of Audio Data on Optical Disks," *Proc. 1st Intl. Conf. on Multimedia Information Systems '91*, Singapore, January 1991, McGraw-Hill, Singapore, pp. 123-134.

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

[19] *Proc. 1st Intl. Workshop on Network and Operating Support for Digital Audio and Video*, Berkeley, CA, November 1990, (ICSI Tech. Rept. TR-90-062).

[20] Anderson, D.P., Tzou, S.Y., Wahbe, R., Govindan, R., Andrews, M., "Support for Continuous Media in the Dash System," *Proc. 10th Intl. Conf. on Distributed Computing Systems*, Paris, France, May 1990, pp. 54-61.

[21] Anderson, D.P., Herrtwich, R.G., Schaefer, C., "SRP: A Resource Reservation Protocol for Guaranteed-Performance Communication in the Internet," *ICSI Tech. Rept.* TR-90-006, February 1990.

[22] Anderson, D.P., Govindan, R., Homsy, G., Abstractions for Continuous Media in a Network Window System," *Proc. 1st Intl. Conf. on Multimedia Information Systems '91*, Singapore, January 1991, McGraw-Hill, Singapore, pp. 273-298.

[23] 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.

[24] Allen, J.F., "Maintaining Knowledge about Temporal Intervals," *Comm. of the ACM*, November 1983, Vol. 26, No. 11, pp. 832-843.

[25] Little, T.D.C., Ghafoor, A., "Multimedia Synchronization Protocols for Broadband Integrated Services," *IEEE J. on Selected Areas in Comm.*, December 1991.

[26] Little, T.D.C., Ghafoor, A., Chen, C.Y.R., "Conceptual Data Models for Time-Dependent Multimedia Data," submitted to the *1992 Workshop on Multimedia Information Systems (MMIS '92)*, Phoenix, AZ.

[27] Ferrari, D., "Client Requirements for Real-Time Communication Services," *IEEE Comm. Magazine*, Vol. 28, No. 11, November 1990, pp. 65-72.

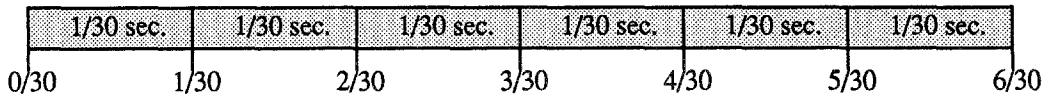

Fig . 1. Temporal-Interval-Based Specification for Video Frames

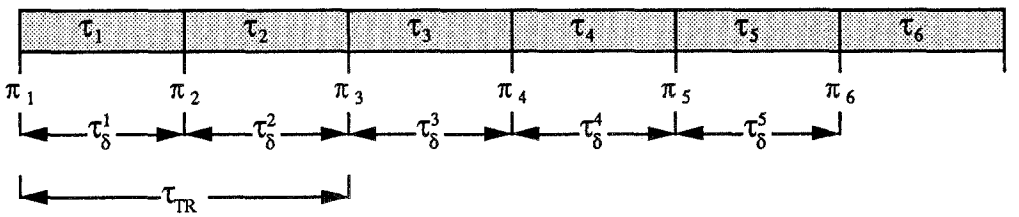

Fig. 2. Timing Parameters

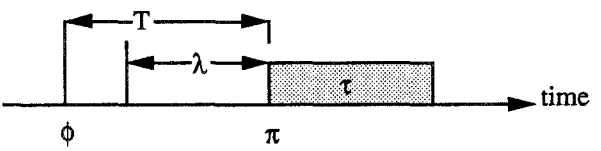

Fig. 3. Timing Parameters Including Latency

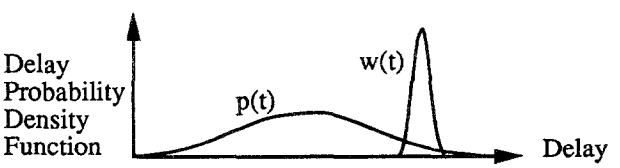

Fig. 4. Reduction of Delay Variance

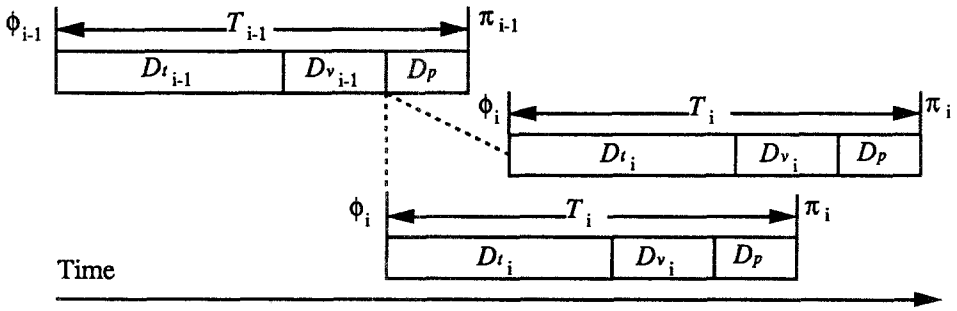

Fig. 5. Scheduling Constraints

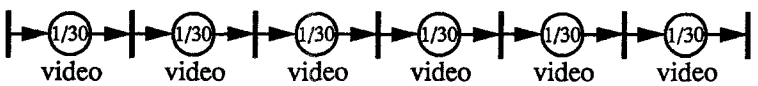

Fig. 6. Video Frame Timing Using the OCPN

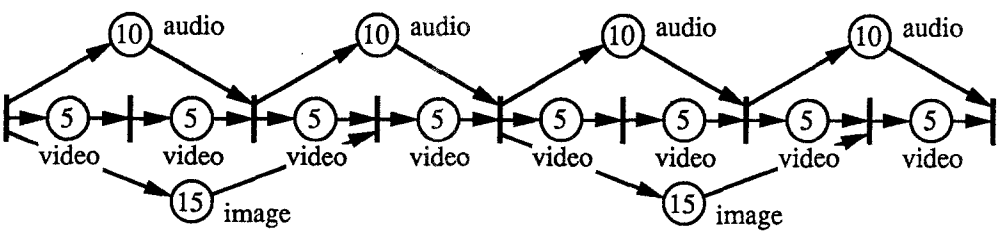

Fig. 7. Audio, Video, and Image OCPN Timing Specification