A Real-Time Synchronization Model and Transport Protocol for Multimedia Applications

Chun-Chuan Yang and Jau-Hsiung Huang

Communications and Multimedia Laboratory
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, R.O.C.

Abstract

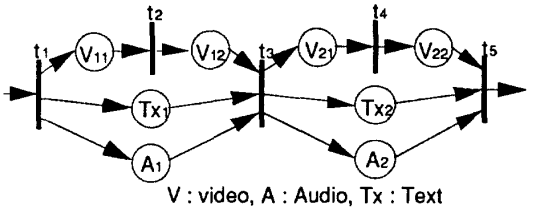
In considering transport layer protocol for multimedia applications in packet networks, real-time and synchronization characteristics of multimedia applications have to be considered. Synchronization means that inter-medium temporal relationship for an application has to be satisfied, after the transmission across the network. Since the random delay of packet networks makes synchronization among media more complicated; therefore, a model is required to specify the temporal relationship among media across the network and can be easily implemented in a transport protocol. In this paper, a real-time synchronization model (RTSM), is presented for synchronization, and a new protocol, namely MSTP (Multimedia Synchronization Transport Protocol), is proposed based on RTSM.

I. Introduction

With the increasing power of modern computers and high-speed network technologies, the integration of multiple media into a single network application [1-8] becomes possible. Multimedia communication applications, such as video phone system, video conference system, shared workspace applications, etc., will lead to a new type of applications by computer and communication. Combining multimedia and communication into computer systems also introduces new research problems. Two of these problems are : (1) *synchronization model* to specify the inter-medium temporal relationships, and (2) *transport protocol* to support multimedia synchronization for real-time communication applications. In designing the synchronization model and the transport protocol, *ease of implementation* and the *non-deterministic behavior* of packet networks (eg., random delay and packet dropping) should receive more attention, which are the major focus of this paper.

Much work [9-13] has been done in describing the synchronization model for multimedia applications. Among those, petri-net based models provide a good method to specify temporal relationships. The modified petri-net model, Object Composition Petri Net (OCPN) model [10-11], has been proved to have the ability to specify any arbitrary temporal relationship among media and it may be applied to both stored-data applications and live applications. The eXtended OCPN (XOCPN) [12] can additionally specify the communication functions. In [13], user actions are also considered. However, when taking the real-time issue of multimedia and the non-deterministic behavior of packet networks into consideration, OCPN/XOCPN and other Petri net based models are not sufficient to deal with the *late transmission* of packets. Here, we define *late transmission* of a packet to be that the packet fails to reach its destination in time (i.e. exceeding its real-time constraint) and should be dropped. We explain the insufficiency of OCPN by the example shown in Figure 1 in the following.

We may treat the example in Figure 1 as an MTV-like application in which there are music (audio) and corresponding video frames and text (subtitle) on the screen. In Figure 1, we only show two cycles of a cyclic model of the application and the interpretation of transition t1 to t3 in Figure 1 follows. Simultaneously, the application plays audio A1, displays two video frames V11, V12 orderly, and displays text Tx1. Figure 1 shows that transition t3 will be fired only after V12, A1 and Tx1 are all played and their durations are expired. Similarly, transition t5 will be fired only after V22, A2, and Tx2 finish playing.



V : video, A : Audio, Tx : Text

Figure 1. An example of petri net based synchronization model.

In the following, we examine the characteristics of audio, video, and text data. For audio data, it is very jitter sensitive and cannot tolerate random delay between audio segments; hence, by holding A2 too long after finishing A1 will result in unrecognizable audio quality. For video

0743-166X/94 $3.00 © 1994 IEEE

data, by dropping a small portion of video frames or by having a small random delay between video frames may only have minor impact on human eyes and is acceptable. For text data, it has the least jitter sensitivity compared to audio and video. Based on such an observation, we notice that audio data is much more jitter sensitive than video and text data and is less tolerant to random transmission delay.

In Figure 1 based on OCPN model, if A1 and Tx1 have finished playing but V12 is late due to network congestion, transition t3 can not be fired until V12 arrives and finishes playing. For this late transmission of V12, A2 can not be played even if it has arrived. However, this is improper since audio is very jitter sensitive. Therefore, a natural thought in handling this situation is to fire transition t3 right after finishing A1 to activate A2, V21, and Tx2 in order to maintain the audio quality. Clearly, the unfinished video frames in V12 will be dropped. Since any of the earlier proposed petri net models does not allow firing transition t3 before finishing V12; hence, a new model will be proposed in this paper and is named as Real-Time Synchronization Model (RTSM).

As to the part of transport protocol, many works [14-20] have been done in discussing high-speed transport protocols. The goals of these protocols are focused on high-throughput or low-latency. The issues of multimedia and real-time synchronization are rarely addressed. In [23], Huitema and Dabbous discussed the synchronization relationship among different types of data. However, that model does not address the real-time characteristics of media, and it can not achieve more complex synchronization relationship for it has only two types of data (RT, DT). In this paper, MSTP is presented to support more general services.

The remainder of this paper is organized as follows. In section II, we describe the RTSM model and the concept of *key medium* and *time medium*. The goal of key medium or time medium is to handle situations when data must be dropped due to late transmissions. Section III describes the architecture of MSTP. The synchronization control and implementation methods of MSTP are presented in section IV. Section V concludes this paper.

II. Real-Time Synchronization Model (RTSM)

In this section, we will present the RTSM synchronization model and the concept of key medium and time medium.

A. RTSM model

Considering the example in Figure 1, in order to avoid the unexpected blocking due to late transmissions, we need a mechanism to enforce the blocked transitions to fire under the situation of late transmissions. Therefore, we introduce a new type of places called *enforced places* to achieve this enforcement. To differentiate an enforced place from a regular place, a double circle is drawn for enforced places as shown in Figure 2. In Figure 2, transition t3 is fed by two regular places, V12 and Tx1, and one enforced place, A1. The rule about enforced place is that once an enforced place gets unblocked (i.e. finishes playing), the transition following it will be immediately fired regardless of the states of other places feeding this transition. Therefore, if A1 becomes unblocked in Figure 2, transition t3 will be immediately fired regardless of the states of V12 and Tx1, and actions of all obsolete places are cleared. In this way, the synchronization anomaly due to late transmissions can be solved.

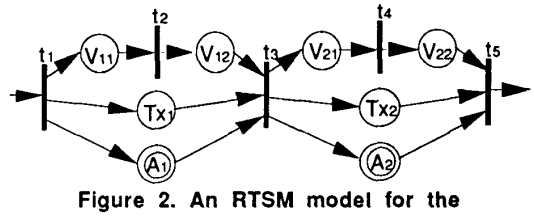


Figure 2. An RTSM model for the example in Figure 1.

B. Concept of Key medium and Time medium

(1) Key medium

In the previous paragraph, we described the special feature of enforced places. In this subsection, we will discuss how to assign enforced places to packets of different media. By definition, enforced places control the advance of time of multimedia applications. That is, if the places of a specific medium, such as voice, are assigned as enforced places, then this medium will not be blocked due to late transmissions of other media in synchronization. Further, this medium will drive other media to advance in time if late transmissions happen to other media. We denote this specific medium as the *key medium*. The concept of key medium is similar to that of master/slave policy proposed in [24], but is more compact for network applications.

Therefore, two factors should be considered in deciding the key medium. First is the importance of the medium and second is the jitter-sensitivity of the medium. For instance, if the quality of one medium is more important than that of others, then this medium should be chosen as the key medium. Or, if the importance of all media are the same, then the most jitter sensitive medium should be chosen as the key medium.

Note that all media except the key medium are subject to packet dropping due to late transmissions. Hence, if there is one medium requiring error free transmission, such as text or numerical data, then this medium should be either chosen as the key medium or cannot be controlled by another key medium. Figure 2 shows an example

where text data and video data can be dropped due to late transmissions when audio data gets unblocked. Hence, in this case, error free transmission is not guaranteed to text and video transfer.

(2) Time medium

Since all media are transmitted across the network, it is also possible for key medium to suffer long delay to exceed its real-time constraint. For example, what should be done if A_1 in Figure 2 is still waiting for its corresponding packets (due to late transmissions or packet dropped in the network) while the real-time constraint of A1 has expired? In such a case, it is reasonable to fire transition t3 to activate A2, V21 and Tx2 instead of waiting for A1 to complete. To achieve this function, the introduction of key medium is not sufficient and we need a mechanism to express absolute time (the real-time constraint) in the model. For this goal, we introduce *time* as one of the media, named *time medium*, and can be placed in the synchronization model of multimedia applications.

Clearly, time medium is a *virtual medium* and normally contains a deterministic duration of time which specifies the real-time constraint between two transitions. Note that time medium can be assigned as regular places or enforced places. By assigning the places of time medium to be enforced places means that the transition following the enforced place of the time medium will be fired when the duration of time specified by the time medium expired, even if the places of other media are still blocked. With the introduction of time medium, we can model the example of Figure 2 to be that shown in Figure 3. Note that in Figure 3, both places of audio and time are assigned as enforced places. Figure 3 shows that transition t3 will be fired if either one of the following two cases are true. One is that A1 has finished playing and becomes unblocked; or, 200 ms has passed by after the firing of transition t1. Hence, if A1 is blocked for a long time due to late transmission, then A1 will be skipped when time medium T1 become unblocked. This resolves the problem of late transmissions of key medium.

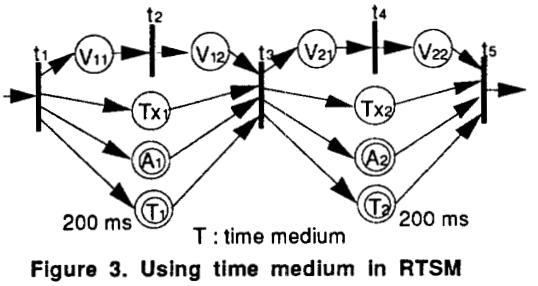


Figure 3. Using time medium in RTSM model.

III. The Architecture of MSTP

A. Basic Concept

MSTP adopts a multi-protocol architecture [22] as shown in Figure 4. For an application with several media, MSTP manager sets up a single multimedia connection, and for different media, it employs different sub-protocols to set up sub-connections with its own quality of service respectively. Each sub-protocol manager manages a sub-protocol connection, which is a transport layer connection, and the function of MSTP manager is to keep track of the operation of all sub-protocol connections such as the set-up and release of each sub-protocol connection. Note that all sub-protocol connections managed by a single MSTP manager belong to a single multimedia application. The purpose of MSTP is to provide a synchronized data streams from sender to receiver, i.e. the sender sends data according to some synchronization relationships and MSTP will deliver data to the application on the receiver conforming to the original synchronization relationships.

Briefly speaking, MSTP manager of the sender accepts RTSM and QoS for each medium from the application program, and sets up a multimedia connection with the MSTP manager of the receiver as mentioned above. After the connection is set up, MSTP manager of the sender will send packets for the application according to RTSM, and MSTP manager of the receiver will deal with out-of-synchronization problems and deliver proper packets to the receiver application. Details of the synchronization control will be provided in section IV. In this way, applications of both sides do not need to deal with the control for synchronization because MSTP managers will handle it. This simplifies the work of the applications.

B. Interactions between MSTP manager and sub-protocol managers

We classify sub-protocols into two categories, with error control and without error control. Sub-protocols which use acknowledgement and retransmission mechanisms for packet loss belong to protocols with error control. Sub-protocols without error control will not retransmit lost packets since there is no acknowledgements. Actions of sub-protocol managers, such as send, receive, acknowledge and discard packets, are under the control of MSTP manager.

For a sub-protocol manager with error control, it accepts send requests from MSTP manager and send packets with proper header values to the corresponding sub-protocol manager of the receiver. When a packet reaches the receiver, the sub-protocol manager will deliver it to MSTP manager immediately and send an acknowledgement for that packet back to the sender. Sub-protocol manager of the sender will retransmit lost packets (using selected repeat discipline). However, MSTP manager can enforce sub-protocol managers to ignore the

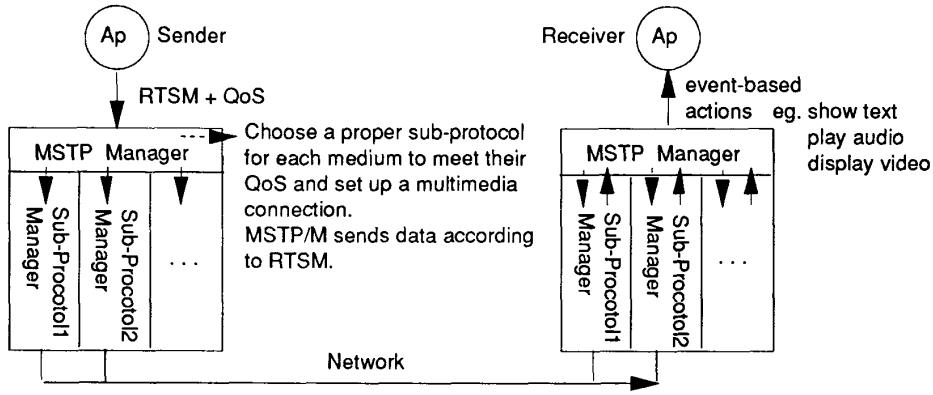


Figure 4. The architecture of MSTP.

lost of some packets by resetting the sequence number of acknowledgements for synchronization purpose. We explain this by an example. In Figure 5, although MSTP manager detects the lost of packet 3, it requires the sub-protocol manager to acknowledge up to packet 4 because the retransmission of packet 3 will be invalid due to its real-time constraint.

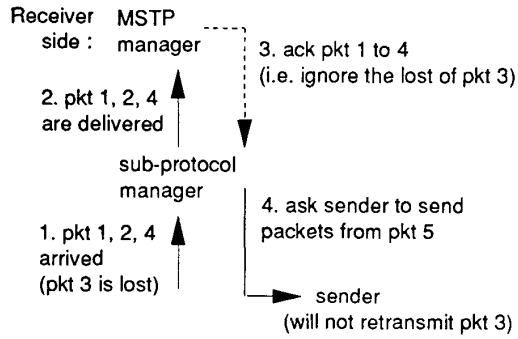


Figure 5. An example of which MSTP manager controls actions of sub-protocol

For a sub-protocol manager without error control, the receiver simply accepts requests from MSTP manager to discard arrived packets which are out-of-synchronization since the sub-protocol dose nothing about error control.

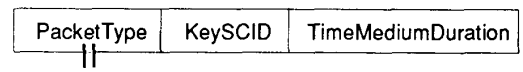
The selection of sub-protocols depends on the QoS of each medium. In MSTP, the selected sub-protocol for the key medium must receive the best service provided by lower layers than those of other media.

IV. Synchronization control in MSTP

In the last section, we described the basic concept and architecture of MSTP. In this section, we will explain how MSTP works to support the synchronized transport services. Note that MSTP manager of the receiver dose not need to know the original RTSM, but deals with synchronization control according to the information contained in packet headers sent by MSTP manager of the sender. Packet format for MSTP and actions which are taken by MSTP manager in the sender and receiver will be presented in the following.

A. Packet format of MSTP

There are two kinds of packets sent by MSTP manager: control packets and data packets. Control packets are used to send control information to set up, release and change configuration of connections. The synchronization-related configurations are : (1) which sub-connection is the key medium sub-connection (denoted by KeySCID), and (2) if the key medium is accompanied with a time medium, what is the duration of the time medium (denoted by TimeMediumDuration). MSTP manager of the sender sends control packets to inform MSTP manager of the receiver about current configuration of the connection. Figure 6 shows the format of control packets. Note that the format only addresses in the aspect of synchronization and dose not show other header informations such as port numbers, QoS etc.. The control packet for the example in Figure 3 is shown in Figure 7 which has to be sent to MSTP manager of the receiver at the phase of connection setup.



C : control KeySCID = Key Sub-Connection ID

Figure 6. The format of the control packets.

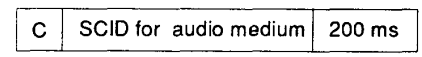


Figure 7. The format of the control packet for example in Figure 3.

The packet format of data packets is shown in Figure 8. The field SCID is used to indicate which sub-connection dose this packet belong to. The KeyRefSeq field indicatesthe sequence number of the key medium packet to

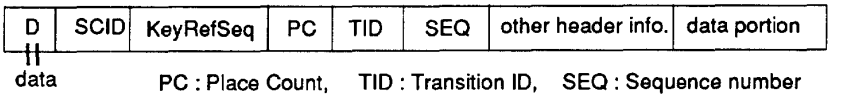


Figure 8. The format of the data packets.

A1	D	1	x	x	x	1	Other	data
Tx1	D	2	1	x	x	1	Other	data
V11	D	3	1	1	t2	(1,1)	Other	data
V12	D	3	1	x	x	(1,2)	Other	data
A2	D	1	x	x	x	2	Other	data
Tx2	D	2	2	x	x	2	Other	data
V21	D	3	2	1	t4	(2,1)	Other	data
V22	D	3	2	x	x	(2,2)	Other	data

SCID of audio = 1, SCID of text = 2,
SCID of video = 3, x = null

Figure 9. Data packets of the example in Figure 3.

which this data packet has to be synchronized with. Notice that the KeyRefSeq fields of key medium packets is null. Fields PC and TID are used for packets of places which do not feed into the same transition with the key medium, e.g. V11 and V21 in Figure 3. Field PC indicates the total number of places that feed into the transition specified in the field TID. For example, in Figure 3, the packet containing V11 will have PC=1 and TID=2; which means transition 2 is fed by only one medium. Hence, transition X will be fired when PC number of packets with TID=X have arrived. Field SEQ indicates the sequence number of this packet. Other header information includes port numbers, QoS, etc.. The data portion field contains the real data of the packet. We again use the example in Figure 3 for illustration. The data packets for the example are shown in Figure 9 in which we use a two-dimensional form for SEQ of video packets for compactness. Note that fields PC and TID for Tx1, Tx2, V12, and V22 are null in Figure 9 since they feed into the same transitions with the key medium.

B. Actions for MSTP manager of the sender

After the multimedia connection is set up, the sending process of MSTP manager of the sender is very simple. As mentioned in section III, the MSTP manager sends packets according to RTSM specified by the application program. It traverses the RTSM and sends packets of the place being visited [11] by making a send request to the corresponding sub-protocol of the packet. Key medium packets have to be sent before other medium packets. The sending process of the example in Figure 3 can be easily realized as shown in Figure 10.

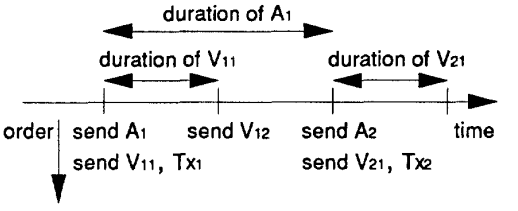


Figure 10. Sending process of the example in Figure 3.

C. Actions for MSTP manager of the receiver

After the multimedia connection is set up, MSTP manager of the receiver knows the current configuration of the multimedia connection, i.e. KeySCID and TimeMediumDuration, according to the control packet sent by MSTP of the sender. MSTP manager of the receiver maintains a set of status variables for each connection for synchronization purpose. These variables (in italic font) include :

(1) two variables to save current configuration : *KeySCID* and *TimeMediumDuration*,
(2) *CurrentKeySeq* to indicate the sequence number of the current active key medium packet,
(3) *NextKeySeq* to indicate the sequence number of the next key medium packet to be received,
(4) a set of variables to keep the sequence number of next-to-be-received packet for each sub-protocol, *Nxt-SCID*.
(5) pairs of variables (*APC*, *TID*) to store the number of packets which feed into transition *TID* in RTSM. *TID* is the ID of the transition, and *APC* is the number of arrived packets whose TID = *TID*.

Besides, a timer is needed for the use of time medium. Initially, *CurrentKeySeq* is null, and *NextKeySeq* is the starting sequence number of the key medium packet. *Nxt-SCID* is initialized to the starting sequence number minus one of the subconnection *SCID*. When MSTP manager receives one packet from the sub-protocol, it will execute proper actions according to the synchronization algorithm shown in Figure 11.

The algorithm uses variable *CurrentKeySeq* to store the sequence number of current active key medium packet, and the value of *CurrentKeySeq* will not be changed until the next key medium packet is received (or, a timeout event occurs). Therefore, any non-key medium packet with KeyRefSeq = *CurrentKeySeq* arrives before the value of *CurrentKeySeq* is changed meets the synchronization

relationship and can be accepted and delivered to application program (Figure 11, case a). The null value of *CurrentKeySeq* means the key medium packet has not arrived yet, and any non-key medium packet whose SEQ ≥ its *Nxt-SCID* arrives at that time is buffered (Figure 11, case b) until one key medium packet arrives (Figure 11, case c). If the time medium is used, a timeout event means that the next key medium packet is too late to be accepted, so that we advance *NextKeySeq* and *Nxt-SCID*s by 1.

Although the MSTP manager of the receiver is not aware of the RTSM, there is an implicit RTSM model produced by the rules of the synchronization algorithm shown in Figure 11. In Figure 12, we draw part of the implicit RTSM of the example in Figure 3 at the receiver.

For applications without use of key medium, RTSM model can be easily implemented in MSTP protocol by using fields PC and TID in the packet format.

D. An example

We use a run-time case for Figure 3 to explain the algorithm mentioned above. One arriving pattern and corresponding actions are shown in Figure 13. In this example, late transmissions of both key-medium and non-key media are considered. The comments of actions taken are provided in the figure.

V. Conclusion

In this paper, we consider the impact of random delay in packet networks on synchronization among media of multimedia applications. The contributions of this paper are listed in the following.

(1) An RTSM with enforced places to deal with late transmissions is proposed.

(2) The concept of key medium in RTSM for synchronization control is suggested.

(3) The concept of time medium associated with key medium to deal with late transmissions of key medium is also suggested.

(4) An implementation method of RTSM in transport protocol MSTP to achieve synchronization service is provided.

Using the transport protocol MSTP, the receiver does not need to know the original RTSM of the sender. Synchronization among media can be easily achieved by sequence numbers provided by MSTP. Hence, the overhead of the transport protocol and the application program can be significantly reduced.

References

[1] C. Nicolaou, "An architecture for real-time multimedia communication systems," IEEE J. Select Areas Commun. vol. 8, no. 3, pp. 391-400, Apr. 1990.

[2] W. H. F. Leung, T. J. Baumgartner, Y. H. Hwang, M. J. Morgan, and S. C. Tu, "A software architecture for workstations supporting multimedia conferencing in packet switching networks," IEEE J. Select Areas Commun. vol. 8, no. 3, pp. 380-390, Apr. 1990.

[3] P. V. Rangan, and D. C. Swinehart, "Software architecture for integration of video services in the etherphone system," IEEE J. Select Areas Commun. vol. 9, no. 9, pp. 1395-1404, Dec. 1991.

[4] K. Maeno, S. Sakata, and T. Ohmori, "Distributed Desktop Conferencing System (MERMAID) Based on Group Communication Architecture," Proc. IEEE ICC, pp. 520-525, 1991.

[5] S. Masaki, H. Yamaguchi, Y. Hayashi, T. Nishimura, and K. Shimamura, "Multimedia Handling Scheme in a Groupware System for B-ISDN," Proc. IEEE GLOBECOM, pp. 747-751, 1992.

[6] M. Arango, P. Bates, M. Cochinwala, D. Cohrs, R. Fish, ... etc, "The Touring Machine System", Comm. ACM, pp. 68-77, Jan. 1993.

[7] Y. V. R. Reddy, K Srinivas, V Jagannathan, and R Karinthi, "Computer Support for Concurrent Engineering," IEEE Computer Mag. pp. 12-15, Jan. 1993.

[8] J. H. Huang, C. C. Yang, W. H. Tseng, C. W. Lee, B. J. Tsaur, L. K. Chuang, and W. J. Liu, "Design and Implementation of Multimedia Conference System on Broadcast Networks," Proc. IEEE LCN 1993.

[9] R.Steinmetz, "Synchronization properties in multimedia systems," IEEE J. Select Areas Commun. vol. 8, no. 3, pp. 401-412, Apr. 1990.

[10] T. D. C. Little, and A. Ghafoor, "Synchronization and storage models for multimedia objects," IEEE J. Select Areas Commun. vol. 8, no. 3, pp. 413-427, Apr. 1989.

[11] T. D. C. Little, and A. Ghafoor, "Multimedia synchronization protocols for broadband integrated services," IEEE J. Select Areas Commun. vol. 9, no. 9, pp. 1368-1382, Dec. 1991.

[12] N. U. Qazi, M. Woo, and A.Grafoor, "A Synchronization and Communication Model for Distributed Multimedia Objects," Proc. ACM Multimedia 93, pp. 147-155.

[13] B. Prabhakaran, and S. V. Raghavan, "Synchronization Models for Multimedia Presentation with User Participation," Proc. ACM Multimedia 93, pp. 157-166.

[14] T. F. Laporta, and M. Schwartz, "Architectures, features, and implementation of high-speed protocols", IEEE Network Mag., pp. 14-22, May 1991.

[15] W. Doeringer, A. Dykeman, M. Kaiserswerth, B. Meiser, H. Rudin, and R. Williamson, "A survey of light-weight transport protocols for high-speed networks", IEEE Trans. Comm. vol. 38, no. 11, pp. 2025-2039, Nov. 1990.

[16] R. M Sanders, and A. C. Weaver, "The Xpress Transfer Protocol (XTP) - A Tutorial", A Quarterly Publication of ACM SIGCOM, vol. 20, no. 5, pp. 67-80, Oct. 1990.

[17] D. R. Cheriton, and C. L. Williamson, "VMTP as the transport layer for high-performance distributed systems", IEEE Comm. Mag. vol. 27, no. 6, pp. 37-44, June 1989.

[18] D. R. Cheriton, "VMTP : A transport protocol for the next generation of communication systems", ACM Computer Comm. Review, pp. 406-415, 1986.

[19] D. D. Clark, M. L. Kambert, and L. Zhang, "NETBLT : A high throughput transport protocol", ACM Computer Comm. Review, pp. 353-359, 1988.

[20] Z. Haas, "A communication architecture for high-speed networking", Proc. IEEE INFOCOM, pp. 433-441, San Francisco, 1990.

[21] J. Kurose, "Open Issues and Challenges in Providing Quality of Service Guarantees in High-Speed Networks," ACM SIGCOM 1992, pp. 6-15.

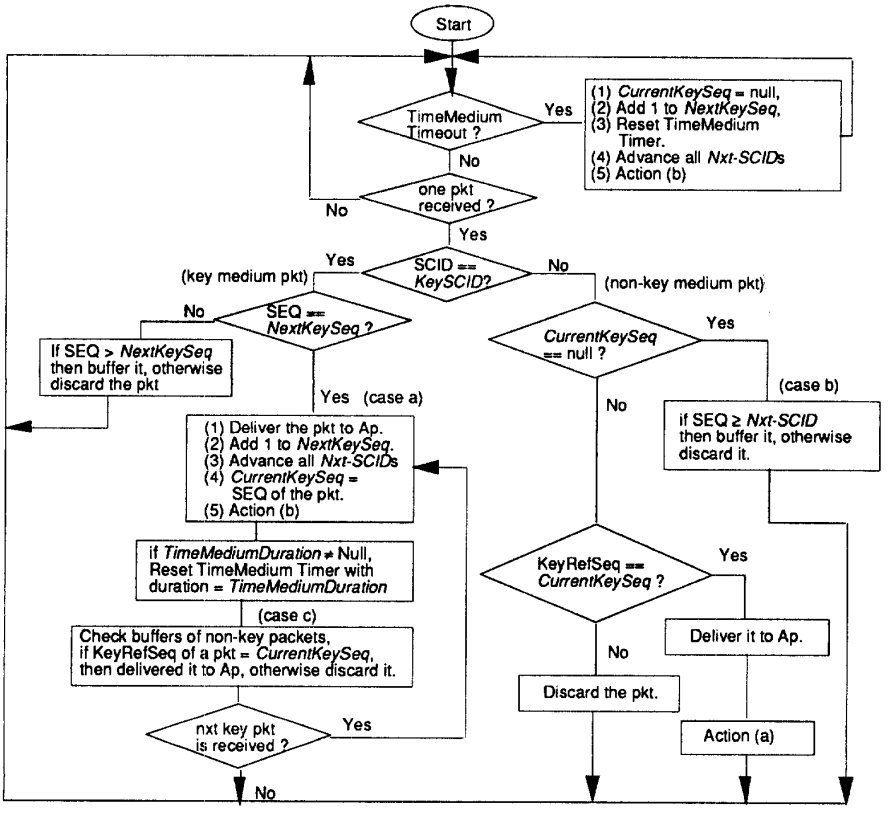
[22] J. H. Huang, and S. H. Lee, "MHTP - A multimedia high-speed transport protocol", Proc. IEEE GLOBECOM, pp. 1363-1368, 1992.

[23] C. Huitema, and W. Dabbous, "Minimal complexity for the simplest protocol", Proc. IEEE INFOCOM pp. 1481-1489, 1991.

[24] D. P. Anderson, and G. Homsy, "A Continuous Media I/O Server and its Synchronization Mechanism," IEEE Computer Mag. pp. 51-57, Oct. 1991.

[25] A. Cambell, G. Coulson, F. Garcia, and D. Hutchison, "A Continuous Media Transport and Orchestration Service," ACM SIGCOM 1992, pp. 99-110.

[26] D. Ferrari, A. Gupta, M. Moran, and B Wolfinger, "A Continuous Media Communication Service and its implementation," Proc. IEEE GLOBECOM pp. 220-224, 1992.



Action (a) : decide if any of TIDs into which no key medium feeds, is ready to fire by checking pairs of (*APC, TID*) if one TID is ready to fire (i.e., pkt's PC = *APC* for the same TID) , advance *Nxt-SCIDs* whose places feed into transition TID

Action (b) : inform sub-protocol managers to accept pkts whose SEQ ≥ NXT-SCID, and pkts beyond this range need not be retransmitted if lost.

Figure 11. The synchronization algorithm of MSTP.

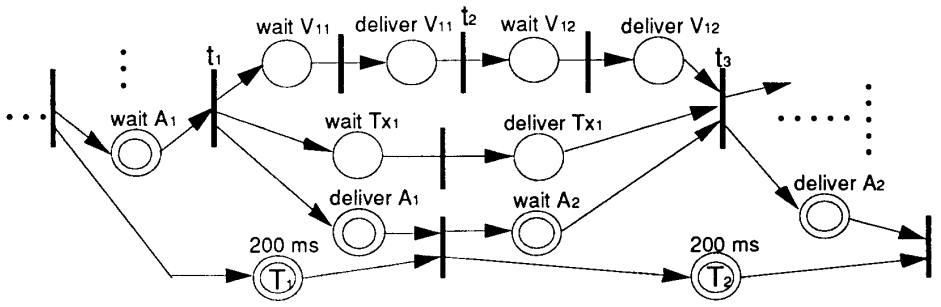


Figure 12. The implicit RTSM of the example in Figure 3 (partial).

	[Packet arrived]				[MSTP variables and actions after pkt arrived]			
time	SCID	KeyRef	SEQ	(PC, TID)	*CurrentKeySeq* *NextKeySeq*	*NXT-2* *NXT-3*	Action	Comments
	1	null	1	null	1, 2	(1,1), 1	accept	
	2	1	(1,1)	(1, t2)	1, 2	(1,2), 1	accept	
	1	null	2	null	2, 3	(2,1), 2	accept	(A)
	3	2	2	null	2, 3	(2,1), 2	accept	(B)
	2	1	(1,2)	null	2, 3	(2,1), 2	reject	(C)
200 ms timeout					null, 4	(3, 1), 3	(D)	(E), (F)

1 = KeySCID of voice with TimeMediumDuration = 200ms, 2 = SCID of video, 3 = SCID of text
NXT-2(3) : sequence number of the next pkt of SCID=2(3)

(A) pkt with SEQ=(1,2) of SCID=2, and pkt with SEQ=1 of SCID=3 dropped due to out-of-synchronization, advance *NXT-2* and *NXT-3*.

(B) text pkt with SEQ=2 accepted, but *NXT-3* is not increased until the next key pkt (A_3) arrives or timeout.

(C) video pkt (1,2) is rejected due to late transmission.

(D) pkt with SEQ=4 of KeySCID=1 dropped. *NextKeySeq*, *NXT-2* and *NXT-3* are advanced.

(E) due to exceeding the duration of the time medium

(F) pkts of SCID=2,3 with KeyRefSeq=3 dropped

Figure 13. A run-time case for example in Figure 3.