Continuous Media Communication with Dynamic QOS Control Using ARTS with an FDDI Network

Hideyuki Tokuda* Yoshito Tobe** Stephen T.-C. Chou* José M. F. Moura**

School of Computer Science
** *Department of Electrical and Computer Engineering*
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213
{hxt,tobe,stc}@cs.cmu.edu

Abstract

Continuous media communication requires timely delivery of data such as digital video and audio packets. Quality of Service (QOS) parameters specify the *temporal* and *spatial* characteristic of such continuous media data. To insure timely delivery of continuous media data, the system needs to minimize the communication delay by securing required processor and network resources. We have extended the Capacity-Based Session Reservation Protocol(CBSRP), which was proposed for realizing predictable real-time communications, to support dynamic control of QOS. We have implemented a QOS control scheme by which the network dynamically adjusts the allocation of network bandwidth on a Fiber Distributed Data Interface(FDDI) network.

In this paper, we describe the definition of our QOS model, the extension of CBSRP to support dynamic control of QOS, and the implementation and evaluation of extended CBSRP in our distributed systems testbed, Advanced Real-Time System (ARTS).

setcountersection0

1. Introduction

Many modern workstations are equipped with specialized hardware capable of producing digital audio and displaying high resolution graphics. One of the current trends in multimedia computing is toward incorporating full motion digital video and audio into applications.

We call these multimedia data communication as *continuous media communication.* In a conceptual model, each data such as a video frame and voice packet must be presented in a continuous fashion at the destination site. A communication network must be able to deliver the continuous media data timely fashion so that the system will not produce any observable jittering effect. We view such a continuous data stream as a sequence of data with its timing constraints attached to each data item. If each data item has the same deadline, we call it *periodic data stream*; otherwise we call *aperiodic data stream.* For instance, we can define real-time video and audio data as a periodic data stream and on-line desktop slide presentation as an aperiodic data stream.

Quality of Services (QOS) in continuous media can be expressed in terms of *temporal* and *spatial* resolution. The temporal resolution is the frame rate or period of the data. When a sequence of continuous media is collected at a microphone or a video camera, the temporal resolution corresponds to the sampling rate. The spatial resolution is associated with the data size or compression ratio. The temporal and spatial characteristics of the periodic and aperiodic data streams are mostly application dependent. However, some generalization can be drawn by observing some continuous media applications. Large volume data are produced and sampled at high rate.

A running continuous media applications, such as full motion video, can occupy significant bandwidth of the computer resource. For example, a movie quality pictures are generated 30 frames per second. A single 640x480 of uncompressed video frame can easily take up to 900KBytes. At this rate, over 26 Megabytes of the pictures must be processed within the 33 millisecond deadline. Although some compression schemes such as JPEG[20], MPEG[7], and px64[10] been suggested

---

*This research was supported in part by the U.S. Naval Ocean Systems Center, the Office of Naval Research, the Federal Systems Division of IBM Corporation, and the SONY Corporation. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of NOSC, ONR, IBM, SONY, or the U.S. Government.

COMM'92-8/92/MD,USA

to reduce the data size, high quality video frames are usually too large for a conventional local area network environment.

For isochronous application such as interactive video/teleconferencing, end-to-end delay has to be bounded and any observable jittering effect should be avoided. Because of these temporal and spatial constraints, continuous media communication remains within "window of scarcity"[1] and thus requires special resource management. This is further complicated by the fact that the system resources, which includes network bandwidth, processor execution time, and communication buffers are shared among many other tasks in the system.

In order to overcome such spatial and temporal constrains of continuous media communication, a few transport protocols such as ST II (Stream Protocol II)[5], SRP (Session Reservation Protocol)[1], and fast lightweight transport protocols, VMTP (Versatile Message Transaction Protocol)[2], XTP (Express Transport Protocol)[14], have been proposed. In general, we can divide these protocols into two classes: *reservation-* and *non-reservation-based* protocols. The ST II and SRP protocols reserve the necessary set of system resources such as processor execution time, buffers, and network bandwidth, before transmitting any data. A similar resource reservation model, a real-time channel, has been proposed for a wide area network environment[6]. The reservation of such resources requires significant operating system and network supports. VMTP and XTP, on the other hand, the data items are transferred best-effort basis and without any resource reservation. However, no guarantee on the end-to-end delay or jitter bound for a session can be made.

We have developed, CBSRP, Capacity-Based Session Reservation Protocol (CBSRP)[3] in order to provide guaranteed end-to-end delivery of data through resource reservation in a local area network environment. CBSRP differs from ST-II and SRP in its capability of changing quality of service (QOS) parameters of a session dynamically. Each unidirectional communication entity for continuous media is called a *session.* CBSRP determines whether or not a session can be created which meets the user's requested QOS parameters. If the minimum QOS specification can be met, the session is established. Quality of service is guaranteed unless a shortage of resources is caused by the introduction of more urgent system tasks.

User demands can be flexible. For example, Some users accept only high quality video, while others are satisfied with lower quality when the system capacity cannot accommodate them otherwise. Some users allow service degradation as long as a specified and agreed upon minimum quality is guaranteed. The system therefore should be adaptable to accommodate various user's QOS requirements.

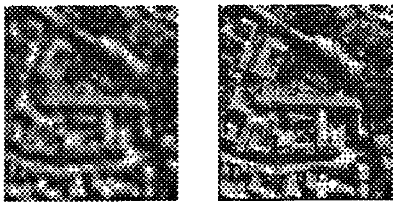
Figure 1: Spatial Resolution of video image

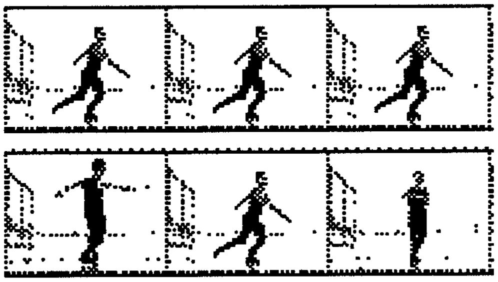
Figure 2: Temporal Resolution video image

In CBSRP, the user can create quality of service classes which specifies acceptable QOS parameters as a discrete function of resource load. The allocation of these parameters may be changed dynamically. When a new request for creating a session comes in, if the system resources are already exhausted, some current sessions may be forced to degrade their QOS parameters to their minimum quality in order to accommodate the new session. The minimum quality of a session is guaranteed once the session is created.

This system has been implemented and demonstrated using a distributed systems testbed, ARTS[17] with an FDDI network. In this paper, we describe dynamic QOS control using CBSRP in ARTS with an FDDI network. In Section 2, we first introduce our model of QOS and discuss a simple specification of QOS and how to control QOS in continuous media communication. We then describe a particular implementation of CBSRP in ARTS with FDDI in Section 3. In Section 4, we show the basic evaluation results of the CBSRP protocol, the conclusion and the future work are mentioned in Section 6.

2. Quality of Service

In this section, we define QOS in continuous media and describe how QOS is controlled in our model.

2.1. Motivation

Quality of services in continuous media can be expressed in terms of *temporal* and *spatial* resolution. Figure 1 shows an example of a difference in spatial resolution for the same video image. A higher spatial resolution carries more accurate data per frame. Temporal resolution, on the other hand, provides a closer sequential match with the original image or data. Figure 2 demonstrates an example of such differences in the same video stream.

These parameters determine the quality of service. For a sender and receiver of continuous media, these parameters are as important as other characteristics which are common to all communications such as end-to-end delay, message order, and reliable delivery of messages. Since user demands can be very flexible, the specification of QOS should be able to providing a variable range of QOS values. The dynamic control of these QOS parameters is also important to adjust a user's demands based on the availability of system resources.

2.2. A Simplified Specification

The user's requested value for temporal and spatial resolution can be quantized in a finite number of QOS classes. Each quality class is a discrete setting of temporal and spatial resolution. We can list classes in a monotonically increasing order based on their system resource requirements. While there may be many such combinations, we usually need only a few classes. These classes specify the gradual degradation of service when the system can no longer guarantee resources for high quality service. We denote C_{tmp} and C_{spt} as the class of temporal resolution and spatial resolution, respectively. A higher order of C_{tmp} could be assigned to high-quality video service, while a lower order C_{tmp} might be assigned to a monitoring system.

For digital video services, the user may choose among period of 1, 2, 3, 5, 10, 15, or 30 frames per second (fps). For digital audio, the number of classes are based on the sampling rate of the audio device. In video applications, the simplest implementation of C_{spt} allocates the number of bits per pixel in proportion to the class, i.e., 8 bits per pixel (bpp) in class 1, 16bpp in class 2, 24bpp in class 3 and so on.

A more sophisticated implementation might first manipulate the image by some signal processing technique. The discrete cosine transform(DCT)[4] is a commonly used method that has been incorporated in the MPEG. Different classes of resolution will result from allocating more DCT coefficients to higher level classes. $p \times 64$ video coding can also utilize C_{spt} with enough classes. In audio applications, for example, where the lowest class is assigned to telephone service and the highest class handles compact disc quality sound, the C_{spt} can be selected in the same way.

The QOS parameters which the user should pass to the system are

- minimum C_{spt}, •maximum C_{spt},
- s_res[MAX_SPT],
- minimum C_{tmp}, •maximum C_{tmp},
- t_res[MAX_TMP],
- Importance,
- allowable end-to-end delay, and
- maximum packet loss rate.

MAX_SPT and MAX_TMP are respectively, the number of classes for the spatial and temporal resolution defined in the system. Let s_res and t_res be the data size of each C_{spt} per one period, and the sampling rate of each C_{tmp}. The value of Importance parameter indicates the order of importance among sessions and is used for deciding C_{spt} and C_{tmp} according to the CBSRP described in the next section. We will focus on how C_{spt} and C_{tmp} are decided.

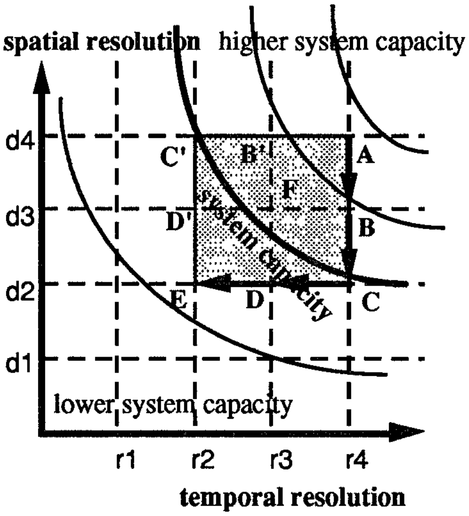

Figure 3: Determination of Resolution

2.3. Control Mechanism

C_{spt} and C_{tmp} play a key role both for deciding the initial QOS and to ensure that the minimum requirements are maintained while the session lasts.

At session creation, the discrete temporal and spatial values for the data transmission are specified in two arrays, s_res[] and t_res[]. A combination of temporal and spatial values defines a system resource requirement for the session. Let D_{user} and R_{user} denote the s_res[C_{spt}] and the t_res[C_{tmp}], respectively. Higher capacity is required for a higher product of D_{user} R_{user}. Therefore admissible C_{spt} and C_{tmp} are determined corresponding to the user's requesting parameters and the capacity of the system.

Figure 3 shows how C_{spt} and C_{tmp} are decided. The product $k = D_{user}R_{user}$ corresponds to the system capacity line in the graph. When t_res[maximum C_{tmp}] =r4, t_res[minimum C_{tmp}] = r2, s_res[maximum C_{spt}] = d4, and, s_res[C_{spt}] = d2, the pair of t_res[C_{tmp}] and s_res[C_{spt}] may be chosen to be any point on or inside the rectangle ACEC' (i.e., point A, B, C, D, E, B', C', D', and C"). Since more than one points can be on the same line, $D_{user}R_{user} = k$, there is no unique way to decide the admissible point. The region on the upper right of the system capacity line indicates higher system capacity. Region on the lower left of the capacity line are of lower capacity. If the returned class is higher than the minimum class, then the session is said to be in an *excess class*. The user can choose temporal-resolution-first (TRF), spatial-resolution-first (SRF), or some hybrid combination of both.

During the session, the C_{spt} and C_{tmp} may be changed dynamically along the grid if there are shortages of available system resources. A session in an excess class can be forced to reduce its C_{spt} or C_{tmp} when the capacity of the system is exhausted and a request for creating a more important session occurs. However, the minimum class E in Figure 3 is guaranteed to be ensured. A user who desires only the maximum C_{spt} and C_{tmp} must set the same value to both the minimum class and the maximum class in each parameter.

3. System Support for QOS

In this section, we will describe how to support QOS in ARTS with an FDDI network utilizing CBSRP.

3.1. Capacity-Based Session Reservation Protocol

CBSRP is designed to minimize variance of delay for stable continuous media communication in a local area network. CBSRP provides guaranteed performance by reserving buffers, processor, and network bandwidth essential for bounded end-to-end communication session. In addition, resource management keeps track of system resource allocation so that no shortage of resources will occur. By preallocating system resources to the time critical activities, we can bound the worst case execution time by removing extra delay and overhead caused by resource contention. CBSRP also allows the dynamic mode change of a session to a higher or lower QOS class in response to the resource requirement of critical tasks. The quality of service which drops during shortage of resources, will be restored to its original class when the system resources become available again.

A *real-time session* is a unidirectional communication path between a sender and a receiver with guaranteed performance. The sender uses an established real-time session and delivers data to a remote receiver object. Each session has a unique session identifier and is registered at both sender and receiver.

Periodic real-time sessions can be distinguished by the periodic interarrival of data to the receiver. In general, messages are delivered through several domains[1] before reaching the receiver: the sender's protocol processing domain, the network domain, and the receiver's protocol processing domain. In order to finish delivery within a bounded delay, processing and delivery at each domain must be met by individual deadlines and the sum of which must be less than or equal to the expected end-to-end delay. If processing within each domain is schedulable, the total delay of end-to-end communication is bounded. We use the deadline monotonic model [9] based on the period and worst case execution time for the schedulabilty check. If a task is schedulable under the deadline monotonic policy, its deadline or delay bound within a scheduling domain can be met (see Section 3.5).

3.2. ARTS and FDDI

ARTS is a distributed real-time operating system being developed at CMU[17, 18]. The objective of ARTS is to develop and verify advanced real-time computing technologies. In ARTS, the basic computational entity is represented as an object[12]. Each object is capable of creating *real-time* or *non-real-time* threads which are explicitly specified with their timing attributes. A real-time thread can be a *periodic* or *aperiodic* thread based on the nature of its activity.

Priority Inversions[16] occur when high priority task blocks while a low priority task is allowed to execute. Priority inversion is common in the time-sharing systems which may result in non predictable system behavior. Communication scheduling delay often encounter more serious priority inversion problems since the protocol modules are multi-layered. ARTS minimize priority inversions within the system and provide foundation for creating predictable real-time sessions[18].

FDDI is a 100 Mbps fiber optics token ring network supported by ARTS. FDDI provides two types of operation: *synchronous* mode and *asynchronous* mode. Since the transmission of synchronous frames take precedence over asynchronous traffic, this allows separation of time critical communication from the non-time-critical traffic. Station can reserve synchronous allocation(SA) to transmit frames in the synchronous mode. With synchronous service, access time can be bounded while the total synchronous transmission time is equal to or less

---

[1] Here, we used a term "domain" for indicating a "scheduling"domain as well as a "resource allocation" domain.

than target token rotation timer(TTRT) minus the overhead. In [15], Sevcik and Johnson proved that when the network operates normally, the token rotation time between two consecutive visits to a node is bounded by 2*TTRT.

FDDI is suitable for real-time communication since it can separate real-time traffic from non-real-time traffic by using the synchronous mode. The synchronous mode traffic is invariant from the non-real-time, asynchronous mode traffic. FDDI could also support the QOS control mentioned above, since it can maintain the synchronous allocation time for each node.

3.3. QOS Conversion

Selecting a reasonable set of QOS parameters for CBSRP is a very important problem. There is no unique way to deliver a set of appropriate parameters for continuous media applications. The QOS parameters described in Section 2.2 were selected in such a way that we could map these parameters into a reasonable set of processor and memory resource related information and lower-level protocol attributes in the system.

For example, QOS conversion from a given frame rate to a periodic transmission of a FDDI frame was computed as follows. The period of the user's video frame stream, P_{user}, which is equal to 1/R_user, and the period of media access control(MAC) P_{MAC} are not always the same. Therefore a per-period data size in MAC, D_{MAC} should be recalculated based on P_{user}, and D_{user}. When FDDI is used, since the maximum length of one frame is 4500 Bytes, fragmentation may be necessary. Each fragmented packet needs D_{proc}(i.e., 56 bytes), a header, and a trailer for protocol processing. Therefore the maximum data size for user data D_{user} should not be more than 4500 bytes $- D_{proc}$.

Once D_{MAC} is calculated, the user's requesting capacity D_{user}/P_{user} can be ordered in a one-dimensional measurement of D_{MAC}. Thus the pair of C_{res} and C_{prd} is converted to a one-dimensional class $C_{session}$. $C_{session}$ and D_{MAC} are used for dynamic QOS control.

3.4. Session Operations

Session creation may be initiated by either the sender or the receiver. Upon invocation, the local session manager reserves the required resources and forward the request to the remote session manager. A session is successfully established only if the required resource are reserved at both sites.

To conform with CBSRP, the sender and the receiver use the library functions briefly described as follows.

rval=**Session_Create**(session_id, mpid, session, abort, relax)
rval=**Session_Close**(session_id)
rval=**Session_Reconfig**(session_id)

```
u_long *session_id;
PID *mpid; /* monitor thread pid */
struct session_dsc *session;
int *abort;
int *relax;

typedef struct session_dsc {
        u_long  sp_id;           /* session id, if known */
        OID     sp_roid;         /* remote s. mgr oid */
        OID     sp_serv_id;      /* sender id */
        ......
        u_long  sp_deadline;     /* suggested deadline */
        ......
        u_short max_s_res;       /* the maximum C_spt */
        u_short min_s_res;       /* the minimum C_spt */
        u_long  s_res[MAX_SPT] ;
        u_short max_t_res;       /* the maximum C_tmp */
        u_short min_t_res;       /* the minimum C_tmp */
        u_long  t_res[MAX_TMP] ;
        u_short present_s_res;   /* present C_spt */
        u_short present_t_res;   /* present C_tmp */
        ......
  } SM_DSC;
```

To create a session, the user specify the host as a set of quality of service parameters. The quality of service parameters specify a range of temporal and spatial combinations which cover the minimum class up to the most desirable excess class. The temporal and spatial constraints can be specified either in a pair of minimum-maximum value or an array of fixed values. The session specification is stored as a session entity record at both the local and the remote host.

3.5. Session and Resource Management

Session Manager

CBSRP is implemented with the following servers: Session Manager(SM), System Resource Manager(SRM), and Network Resource Manager(NRM). SM handles creation, termination, and reconfiguration requests from the users and negotiations with the remote SM. NRM and SRM handles admission control and resource management.

SM, SRM, and NRM are kernel objects. SM and SRM at each host handle all session requests to and from their host. The interactions between a sender, a receiver, SM, and NRM are shown in Figure 6. SRM provides **Buffer_Check** and **Schedulability_Check** services and reports the availability of processor resources to SM upon request.

The functions of SM are described in pseudo code

```
/*
 * Body of Session Manager
 */
while(1) {
  if(Accept(object, message, &param1) < 0) continue;
   else {
   switch(message){
    case Session_Create:
        Calculate $C_{session}$ and its $D_{MAC}$;
        Buffer_Check;
        Schedulability_Check;
        Request(remote_SM, Session_Connect, &param);
        break;
    case Session_Close:
        Dealloc_Resource;
        Request(remote_SM, Session_Abort, &param);
        break;
    case Session_Connect:
        Buffer_Check;
        Schedulability_Check;
        Request(NRM, Network_Add, &param);
        if(there is any session whose class should be reduced)
            Request(sender, Network_Change, &param);
          Include acquired $C_{session}$ in returning values;
          break;
    case Session_Abort:
        Dealloc_Resource;
        break;
    case Session_Reconfig:
        Calculate $C_{session}$ and its $D_{MAC}$;
        Buffer_Check;
        Schedulability_Check;
        Request(remote_SM, Session_Recalc, &param);
        break;
    case Session_Recalc:
        Buffer_Check;
        Schedulability_Check;
        Request(sender, Capacity_Change, &param);
        break;
    case Network_Change:
        Request(sender, Capacity_Change, &param);
        break;
   }
   Reply(object, &rval);
  }
}
```

Figure 4: SM Operations

```
/*
 * Body of Network Resource Manager
 */
while(1) {
if(Accept(object, message, &param1) < 0) continue;
else {
 switch(message){
   case Network_Add:
      class = Max_class;
        while( class >= Min_class ){
          if( BW[class] + Used_BW < Max_BW ){
            Used_BW += BW[class];
            Reply(object, &class);
          }
          else class−−;
        }
        Left_Capacity = Max_BW − Used_BW;
        s_num = 0;
        while( Left_Capacity < BW[Min_class] ){
          s_id[s_num] = Pop_Session_Excess;
          if( Importance[s_id[s_num]] > Importance ){
           Reply(object, &FAIL);
          }
          Left_Capacity += Excess_BW[s_id[s_num]];
          s_num++;
        }
        Used_BW += BW[Min_class] − Left_Capacity;
        for( i=0; i<s_num; i++){
          if( object[s_id] <> object ){
           Request( host[s_id], Network_Change, &param);
          }
          else{
           Include s_id in rval;
          }
        }
        Reply(object, &rval);
        break;
   case Network_Release:
        Used_BW −= BW[class];
        Reply(object, &rval);
        break;
   }
 }
}
```

Figure 5: Bandwidth Control in NRM

in Figure 4. The invocation received by **Accept**[2] is processed according to the type of the operation. **Session_Create, Session_Close,** and **Session_Reconfig** use a simple synchronous protocol. The resources are reserved using a greedy algorithm[3]. As soon as the local system resources are checked and reserved, SM issues to **Session_Connect, Session_Abort,** or **Session_Recalc** operation to the remote host. A session operation is successful if and only if operations at both local and remote hosts are successful. The state of a session will be restored to its previous state if the request cannot be met.

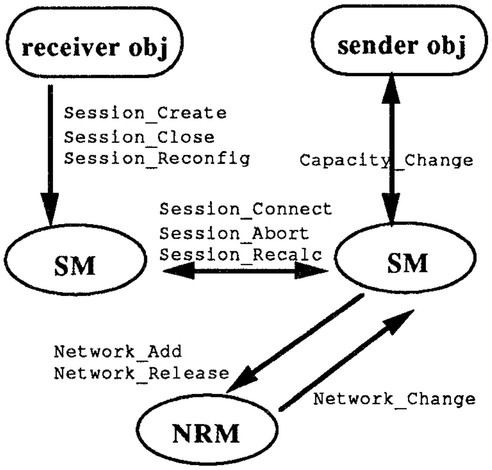

Figure 6: Session Invocations

In order to prevent transient overload in the system, SRM evaluate the schedulability of the task set before creating a new periodic task, aperiodic server, or real-time session, The schedulability test determines the maximum delay bound in the protocol processing domain which can be met. If the task is not schedulable because of overload or unable to meet its deadline, the schedulability test will reject the request, and make SM reduce its QOS class or initiate a negotiation for more resources.

The Session Manager and Network Resource Manager use Real-time Transport Protocol (RTP)[18] for remote invocation. RTP provides reliable, prioritized processing of messages. Because session management does not require guaranteed performance, remote invocations are delivered on a the best effort basis on a non-real-time channel. Non-real-time traffic is sent via FDDI asynchronous service.

Continuous media communication needs a light protocol with high throughput rather than a reliable protocol which provides retransmission. The User Datagram Protocol(UDP)[13] with FDDI synchronous service is thus selected with performance guaranteed by the established session through CBSRP.

Schedulability Test

We consider a set of periodic task, $\tau_1, \ldots, \tau_n$ lists in priority order. Each task is characterized by three components (C_n, T_n, D_n) where

C_i = deterministic computation time of job τ_i
T_i = period of τ_i
D_i = deadline of τ_i

Lehoczky have derived schedulability test for a generalized fixed priority scheduling policy based on the worst case phasing[8]. At the critical instance[11] where all tasks are initiated at same time, all tasks experience the worst response time. For the fixed priority scheduling, if the schedulability test for the critical instant is statisfied, then the task set is schedulable.

For the case where the task's deadline is less than or equal to the period, $D_i \leq T_i$, the schedulability test of τ_m under the deadline monotononic scheduling policy can be simplified to the following.

$$\sum_{j=1}^{m-1} C_j \lceil \frac{D_m}{T_j} \rceil + C'_m \leq D_m \qquad (1)$$

where the summation of τ_1 to τ_{m-1} include all tasks of higher priority and C'_m is the sum of execution time from the same priority tasks. This quantity gives the total cumulative processor demands made by all jobs $\tau_1, \tau_2, \ldots, \tau_{m-1}$ and the same priority level task τ_m during the time $[0, D_m]$.

3.6. Network Resource Management

Network Admission Control

NRM maintains network resources through the *network admission control* and the *network bandwidth* enforcement. The admission control defines a policy of admitting a new reservation request based on the type of the network, traffic patterns, and presently established sessions. The new request is only admitted if the network capacity are available to create the new session without affecting others. The network bandwidth enforcement mechanism monitors the traffic flow into a network and present network allocation.

After receiving the request, NRM tries to assign the highest possible of QOS class from the available network bandwidth. We define

- $C_{i,j}$: the execution time needed to transmit all the data of node i, session j

---

[2] **Request, Accept,** and **Reply** are object invocation primitives in ARTS. Objects invoke the operation of the other object and send data as part of the invocation.

[3] A greedy algorithm makes the locally best choice that looks best at the moment and hope it would lead to an optimal solution in the future. By applying greedy reservation, the system attempts to eliminate jitters caused resource contention in the future.

- $D_{i,j}$: the network deadline of node i, session j
- $L_{i,j}$: the worst case gap between issuing the first and the last packet to the network of node i, session j

In order to create a session, the following criteria must be met: First, the sum of synchronous allocation must be less then TTRT minus σ, the transmission overhead.

$$\sum_{i=1}^{n} SA_i \leq TTRT - \sigma, \tag{2}$$

Second, base on Sevcik and Johnson's derivation, the minimum network deadline of a session must be at least twice of TTRT.

$$D_{i,j} \geq 2 \cdot TTRT \tag{3}$$

In the case of fragmented delivery which breaks data into several packets, the last packet must be delivered at least 2*TTRT before the deadline of the node.

$$D_{i,j} \geq L_{i,j} + 2 \cdot TTRT \tag{4}$$

Third, the synchronous allocation of a node must be able to transmit the packets from all the sessions at least 2*TTRT before the deadline. We assume that all synchronous frames are queued in a FIFO order the FDDI interface unit. In order to guarantee all the deliveries, the network deadline of a node must be set to the minimum network deadline, $D_{i,min}$ among the all outstanding sessions. Thus SA_i must be allocated base on $D_{i,min}$. The following synchronous allocation formula must be satisfied for SA_i for the worst case assumption where all sessions send packets within TTRT.

$$SA_i \geq \frac{\sum_{j=1}^{n} C_{i,j}}{\lceil (D_{i,min} - TTRT)/TTRT \rceil} \tag{5}$$

Given $C_{i,j}$, $L_{i,j}$, and $TTRT$ at node i, we can determine whether all sessions can be admitted, and assign the new SA_i. If unused network bandwidth cannot meet the requirement, NRM searches for a less important session which is operating in its excess class. NRM then forces these sessions to give up their excess allocation. The **Network_Change** message is sent by NRM to the SM whose session is forced to reduce its $C_{session}$.

Network Bandwidth Control

NRM's network bandwidth management is shown in Figure 5. In this figure, BW[class], Max_class, Min_class, and Importance correspond to $C_{session}$ class, the maximum $C_{session}$, the minimum $C_{session}$, and the Importance of the requesting session respectively. These values are sent from SM during **Network_Add** requests. Used_BW is the synchronous bandwidth consumed by all sessions and max_BW is the maximum synchronous bandwidth. Excess_BW[s_id] is the difference between BW[present $C_{session}$] and BW[minimum $C_{session}$] of a session s_id. **Pop_Session_Excess** function returns a session ID from a priority queue of ascending order which maintains the information on sessions on their excess class.

3.7. Example of QOS Control

Figure 7 shows a scenario where a session creation results in renegotiation of resource allocation due to inadequate network bandwidth. (RQ) and (RP) stand for a request and a reply, respectively. A user (receiver2) at node 1 issues a **Session_Create** request to its local SM. After successfully reserving its local resources, the local SM issues a **Session_Connect** request to establish a session. After the remote SM services the request and successfully reserves its local resources, it sends a request to NRM to reserve network bandwidth. If the NRM decides that there is not enough bandwidth available, NRM chooses the lowest priority session (sender1) among outstanding sessions and force it to degrade its quality of service with **Network_Change** request. If the service can be degraded, a request success acknowledgment will be sent back to NRM and from NRM to SM at node 2; session establishment will be sent to receiver1. If any step of the reservation fails, the resources will be freed and the request will be rejected.

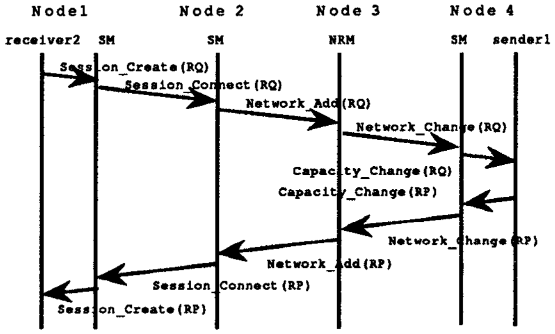

Figure 7: Sequence of Session Creation with QOS Reduction

4. Results

In this section, we compare two cases with and without the network resource manager and show how spatial resolution is controlled at the network. The interframe time of arriving frames is measured using three SONY NEWS-1720 workstations(25 MHz MC68030)

#renegotiations	0	1	2
Session_Create	90.2 ms	118.5 ms	146.5 ms
Session_Connect	74.6 ms	102.6 ms	131.0 ms

Figure 8: Session Creation and Negotiation Cost

Session_Close	65.6 ms
Session_Abort	42.2 ms
Session_Reconfig	71.1 ms
Session_Recalc	55.4 ms

Figure 9: Session Operation Cost

with an FDDI-adapter board SONY IKX-378(AMD SUPERNET Chip set), and a timer board. Workstation are used to send or receive periodic data, or generate background traffic. The timer consists of several counter TTL IC's with an accurate clock on the timer board. We used it for our timing measurements on a NEWS-1720 through a VME-bus backplane. This timer enabled us to measure the interframe times with the resolution of $1\mu s$.

Basic Session Operation Cost

Figure 8 lists the cost of each the session manager's operation with no background traffics. The cost of **Session_Create** includes the cost of buffer check, schedulability test, processor resource reservation, and waiting for remote resource reservation. **Session_Connect**'s cost includes testing, local resource reservation and, request to NRM. Numbers of renegotiations shows how many established session are forced to reduce their classes by NRM. The cost of session creation depends on number of negotiation. For each additional renegotiation increases the session creation time by approximately 28 ms.

Figure 9 shows the cost of session termination and reconfiguration. **Session_Close** and **Session_Abort** free their network bandwidth and processor resources allocation. The cost of **Session_Reconfig** and **Session_Recalc** include the cost of degrading its QOS class.

Spatial Resolution Control

To simulate a traffic of continuous media, two sessions, S_1 and S_2, are created. In each session, the temporal resolution is fixed and only C_{spt} can be controlled by the session managers in order to verify the functions of the managers. The requesting parameters of these sessions are

maximum $C_{spt} = 2$, minimum $Cspt = 0$,
s_res[0] = 8KBytes,
s_res[1] = 16KBytes,
s_res[2] = 24KBytes,
maximum C_{tmp} = minimum $C_{tmp} = 0$,
t_res[0] = 1/(30msec), and
Importance = 10.

To determine the effect of network bandwidth allocation, four cases are compared.

- *Case 1* No background traffic
- *Case 2* With background traffic 1 without NRM
- *Case 3* With background traffic 1 with NRM
- *Case 4* With background traffic 2 with NRM

The requesting parameters of the background traffic 1 are

maximum $C_{spt} = 0$, minimum $C_{spt} = 0$,
s_res[0] = 104KBytes,
maximum C_{tmp} = minimum $C_{tmp} = 0$,
t_res[0] = 1/(10msec), and
Importance = 5,

while the requesting parameters of the background traffic 2 are

maximum $C_{spt} = 1$, minimum $C_{spt} = 0$,
s_res[0] = 64KBytes,
s_res[1] = 104KBytes,
maximum C_{tmp} = minimum $C_{tmp} = 0$,
t_res[0] = 1/(10msec), and
Importance = 12.

A background traffic is generated by the periodic invocation of frames with a dummy destination address. The dummy frames are generated before the traffic is initiated to avoid any overhead of protocol processing. For those four cases, the arriving times of FDDI frames were measured at the receiving host. The background traffic is generated first and then other sessions are created.

We set the requesting parameters to values which will clearly show the difference of each case. TTRT value is set to 15msec and Max_BW is set to 27.5msec which corresponds to 343750 bytes($= 100Mbps/8(bits/byte) * 27.5msec$). The values of Importance are chosen to compare the case in which Importance of S_1 or S_2 is higher than that of the background traffic, and the opposite case. Since D_{umax} is set to 4096 bytes, D_{MAC} of S_1 or S_2 at $C_{res} = 0, 1,$ and 2 are 8304 bytes, 16608 bytes and 24912 bytes, respectively. D_{MAC} of the background traffic 1 and the background traffic 2 at $C_{res} = 1$ are 323856 bytes and 199296 bytes, respectively.

Time 0 is the time the first frame is received at the receiving host. In Figure 10 through Figure 12, ○ indicates the arriving of a 4K-byte S_1 frame and □ indicates the arriving of a 4K-byte S_2 frame. S_1 data and S_2 data are correctly delivered every 30msec in Case 1. However in Case 2, due to the heavy traffic, in excess of the network's capacity, the data are no long delivered every 30msec and the delay increases. This is validated by $323856 + 2 \times 24912 > 343750$. When the NRM is working, Case 3, both S_1 and S_2 are

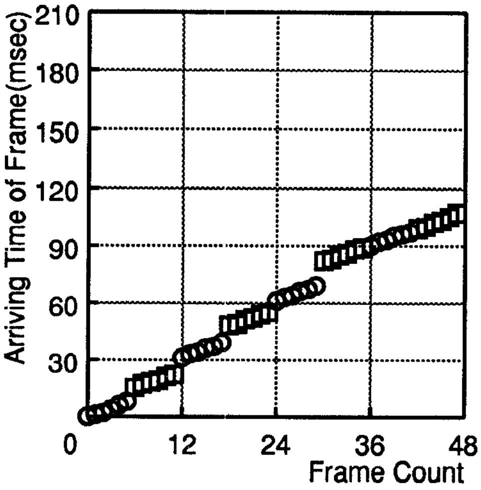

Figure 10: Timing of receiving frames for Case 1

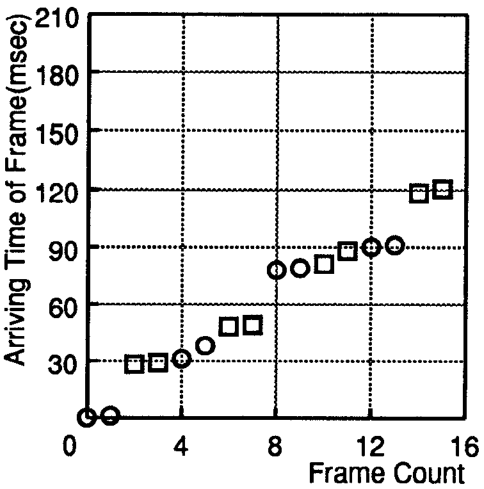

Figure 12: Timing of receiving frames for Case 3

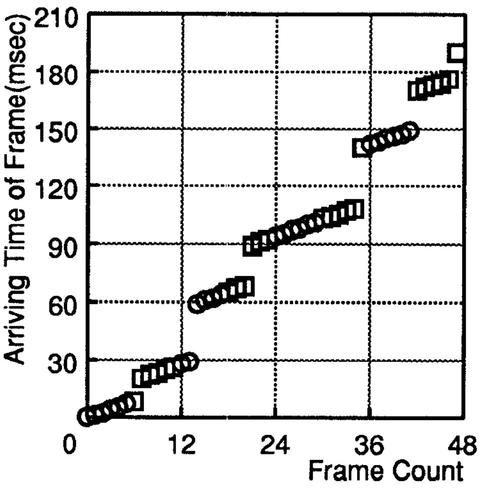

Figure 11: Timing of receiving frames for Case 2

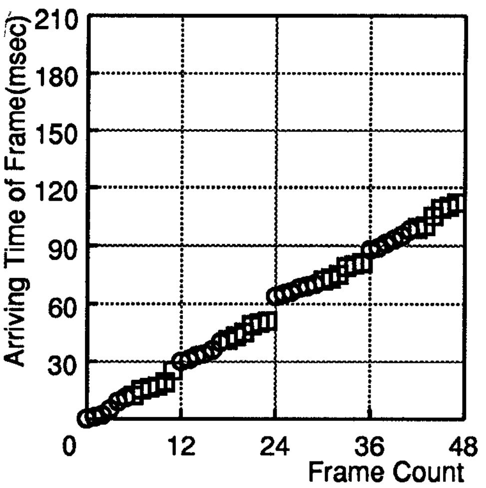

Figure 13: Timing of receiving frames for Case 4

reduced in class, the transmission of both being able to continue every 30msec. The selection of classes are ensured by $323856 + 2 \times 8304 < 343750$ and $323856+8304+16608 > 343750$. In Case 4, the NRM suppresses the class of background traffic so that S_1 and S_2 are able to continue transmission at the maximum class level. The following condition is also ensured; $199296+2 \times 24912 < 343750$.

This example demonstrate the dynamic control of spatial resolution by the NRM. It is important to note that defining spatial and temporal resolution is meaningful only when the system supports the control of the resolution.

5. Conclusions and Future Work

We have demonstrated that the combination of classes of QOS parameters and a dynamic QOS control scheme can facilitate flexible and predictable continuous media transmission in an FDDI network. CBSRP was also effective mechanism for realizing the proposed scheme. While this paper only evaluated dynamic QOS changes with respect to network bandwidth, it is also extensible to other system resources such as processor cycles and system buffers.

Similar to SRP, CBSRP can guarantee a bounded end-to-end delivery service through system resource reservation. However, while SRP requires a static performance parameter for opening a real-time stream and reserving system resources, CBSRP allows a range of QOS parameters which permit service quality to be degraded gracefully and later restored as resource availability varies.

Further evaluation and comparison of TRF, SRF, or hybrid combinations of the two policies must be investigated. We plan to evaluate CBSRP on our RT-Mach[19] platform using the XTP protocol. We will also continue to investigate dynamic control of QOS with video compression and the effect of delay vs. QOS for the continuous media communications.

6. Acknowledgment

We would like to express our appreciation to Stefan Savage for helpful comments and assistance of using his FDDI driver, and Shirish Sathaye for his comment on FDDI synchronous bandwidth reservation. We also thank the ARTS group members for useful discussions.

References

[1] D. P. Anderson, S.-Y. Tzou, R. Wahbe, R. Govindan, and M. Andrews. Support for Continuous Media in the DASH System. Technical report, University of California, Berkeley, October 1989.

[2] D. R. Cheriton. VMTP: Versatile Message Transaction Protocol. Technical report, Computer Science Department, Stanford University, January 1987.

[3] S. T.-C. Chou, H. Tokuda, and M. Morioka. System Support for Continuous Media Communication with Capacity-Based Session Reservation. submitted for publication, 1992.

[4] R. J. Clarke. *Transform Coding of Images*. Academic Press, New York, NY, 1985.

[5] Topolcic et al. Experimental Internet Stream Protocol, version 2 (ST-II), 1990.

[6] D. Ferrari and D. C. Verma. A Scheme for Real-Time Channel Establishment in Wide-Area Networks. *IEEE Journal on Selected Areas in Communication*, 8(3):368–379, April 1990.

[7] D. L. Gall. MPEG: A Video Compression Standard for Multimedia Applications. *Communications of the ACM*, 34(4), April 1991.

[8] J. P. Lehoczky. Fixed Priority Scheduling of Periodic Task Sets with Arbitrary Deadlines. In *Proceedings of the 11th IEEE Real-Time Systems Symposium*, December 1990.

[9] J. Leung and J. Whitehead. On the complexity of fixed-priority scheduling of periodic, real-time tasks. *Performance Evaluation*, 2, 1982.

[10] M. Liou. Overview of the px64 kbit/s Video Coding Standard. *Communication of the ACM*, 34(4), April 1991.

[11] C. L. Liu and J. W. Layland. Scheduling Algorithms for Multiprogramming in a Hard Real Time Environment. *JACM*, 20(1):46–61, 1973.

[12] C. W. Mercer and H. Tokuda. The ARTS Real-Time Object Model. In *Proceedings of 11th IEEE Real-Time Systems Symposium*, December 1990.

[13] Postel. User Datagram Protocol, August 1980.

[14] Protocol Engines, Inc., Santa Barbara, CA. *XTP Protocol Definition, Revision 3.5*, September 1990. PEI 90-120.

[15] K. Sevcik and M. Johnson. Cycle Time Properties of the FDDI Token Ring Protocol. *IEEE Transaction on Software Engineering*, 13(3), 1987.

[16] L. Sha, R. Rajkumar, and J. P. Lehoczky. Priority Inheritance Protocols: An Approach to Real-Time Synchronization. *IEEE Transactions on Computers*, 39(9), September 1990.

[17] H. Tokuda and C. W. Mercer. ARTS: A Distributed Real-Time Kernel. *ACM Operating Systems Review*, 23(3), July 1989.

[18] H. Tokuda, C. W. Mercer, Y. Ishikawa, and T. E. Marchok. Priority Inversions in Real-Time Communication. In *Proceedings of 10th IEEE Real-Time Systems Symposium*, December 1989.

[19] H. Tokuda, T. Nakajima, and P. Rao. Real-Time Mach: Toward a Predictable Real-Time System. In *Proceedings of USENIX Mach Workshop*, October 1990.

[20] G. K. Wallace. The JPEG Still Picture Compression Standard. *Communications of the ACM*, 34(4), April 1991.