ADAPTIVE CONNECTION MANAGEMENT FOR MISSION CRITICAL APPLICATIONS OVER ATM NETWORKS

A. Sahoo, B. Devalla, Y. Guan, R. Bettati, W. Zhao
Department of Computer Science, Texas A&M University, College Station, TX 77843-3112
{asahoo,badari,yguan,bettati,zhao}@cs.tamu.edu

Abstract

This paper focuses on connection management for mission critical real-time applications over ATM networks. Traditional connection management generally requires Quality-of-Service (QoS) parameters to be specified as fixed values, and can only provide a QoS that is constant throughout the lifetime of an admitted connection. Such simplistic specification and consequent resource management offer no flexibility to user applications. The applications cannot receive the best possible QoS, and system resources are grossly under-utilized. We take an adaptive approach. With our adaptive connection management, QoS of connections is specified over a range of values. Resources are reallocated and redistributed in response to dynamic fluctuations in resource availability. With our adaptive strategy, we demonstrate dramatic improvements in both the offered QoS to applications, and the effective utilization of system resources. Our approach is practical and compatible with current networking standards. We have implemented adaptive connection management in a newer version of our real-time toolkit, NetEx. NetEx provides delay guaranteed communication services for mission critical real-time applications over high-speed networks.

1 Introduction

In this paper, we report a new adaptive approach for providing effective and efficient connection management service for mission critical real-time applications. These applications typically consist of a set of tasks executing on different hosts, exchanging messages to co-operatively accomplish a common mission critical objective. Examples of such applications include supervisory command and control of defense systems, manufacturing plants, etc. In addition to logical correctness in execution, they also require timing correctness. The success of a distributed mission critical application thus crucially depends on the ability of the underlying network to guarantee upper bounds on message transfer delay. Our study focuses on connection management for such applications. With our adaptive strategy, we can demonstrate dramatic improvements in both the offered Quality of Service (QoS) to the applications, and the effective utilization of system resources. The results of our study have been implemented on an experimental test-bed consisting of workstations interconnected by an ATM network.

1.1 Traditional Connection Management

We focus on enhancing communication support for mission critical applications through innovative connection management. A connection can be viewed as a *contract* between an application and the connection management system. The defining characteristic of connection-oriented communication is the existence of a connection establishment phase preceding the actual data transfer. Connection management is a network function that is responsible for setting up, maintaining and tearing down connections. A *real-time* connection is additionally characterized by stringent deadline constraints imposed on its packet delivery time. A traditional connection management (TCM) system for real-time connections is shown in Figure 1.

In traditional connection management, an application that requests a new connection issues a Connection Admission Request (CAR) with QoS (i.e., deadline and traffic specifications) needed during the lifetime of the connection. TCM computes the upper bound on the end-to-end delay suffered by the incoming connection, and also recalculates delays of the existing connections. This is because the admission of the new connection may affect the delays of some of the existing connections. TCM then checks if the delays of the incoming and existing connections are less than their respective deadlines. This is to ensure that admitting the new connection does not violate the guarantees made to the existing connections. If delays of all the existing connections and the new connection are less than their respective deadlines, then the new connection is admitted and connection management allocates resources (virtual channels, bandwidth etc.). Data transfer proceeds as a sequence of messages generated according to traffic parameters presented at connection admission time. At the end of the lifetime of a connection, an application sends a Connection Termination Request (CTR) and the TCM releases all resources allocated at admission time.

Much work has been done on traditional connection management. TCM generally requires the QoS parameters be specified as fixed values (e.g., traffic with peak 10 MBPS, and deadline 30 milliseconds). Once a connection is admitted, TCM provides a constant QoS to the connection throughout its lifetime. Such a simplistic specification and consequent resource management suffer from many shortcomings that directly affect the applications using it. Specifically, this model is:

- *Restrictive*: The TCM ignores the fact that for many applications, QoS requirements do not have to be constant. The fixed-QoS model is too restrictive especially for applications that may want to accept admission at a lower QoS, rather than being rejected. For example, a video-on-demand application may be willing to accept a lower QoS (in terms of lesser bandwidth, jitter etc.) to send video frames of poorer quality rather than send no frame at all.
- *Static:* The QoS offered to connections does not change over its lifetime even though the resource availability changes. This precludes connections from receiving the best possible QoS. For example, when a connection leaves the system, existing connections do not benefit from the resources released during connection termination.
- *Rigid:* TCM also lacks the flexibility of providing users with control over the connection admission process. Some applications (e.g., the one that triggers missile deployment), may demand preferential connection admission based on criticality. This necessitates connection management to be able to accept and respond to the directives originating from user level applications.

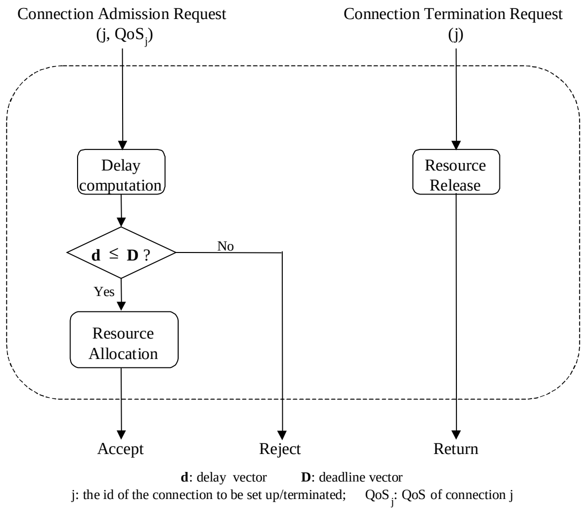

Connection Admission Request
(j, QoSj)
Connection Termination Request
(j)
Delay computation
Resource Release
d ≤ D ?
No
Yes
Resource Allocation
Accept
Reject
Return
d: delay vector
D: deadline vector
j: the id of the connection to be set up/terminated;
QoSj: QoS of connection j

Figure 1: Traditional Connection Management

- *Poor Performance:* A principal measure of any connection management system is its effectiveness expressed in terms of the services provided to applications and utilization of resources. TCM is very ineffective, as it neither exploits the dynamism of the system nor the flexibility in QoS suitable to applications. This also leads to a gross under-utilization of system resources.

1.2 ACM and Issues in Adaptation

To address the shortcomings of a TCM system, we propose Adaptive Connection Management (ACM) for mission critical applications. First, we allow an application to specify QoS in a range, rather than fixed values. Second and more importantly, we incorporate QoS adaptation that offers the best possible QoS to connections contingent on the resources are available.

Our approach has many benefits albeit offering deterministic performance guarantees on end-to-end message transfer delay. The probability of a connection being admitted is increased as ACM has a choice over the level of QoS to offer. The user has control over the admission process by participating in adaptation. ACM allows applications to specify directives to shrink (or expand) QoS of existing connections during admission (or termination). An adaptive resource allocation cognizant of the dynamic fluctuations in resource availability leads to a better utilization of system resources. In addition, at any given time, the existing connections are offered the best possible QoS allowed by resources available. The adaptation mechanism is described in detail in Section 3.

Adaptive connection management is, nevertheless, a challenging proposition. We identify the following important issues and address them in ACM:

- *Efficiency*: To provide delay-guaranteed communication, various decisions have to be made in connection management and resource allocation. It is imperative to make these decisions as fast as possible because mission critical applications demand a fast response for their requests. This requires minimizing the overhead in the decision making while not compromising the quality of service provided.
- *Effectiveness*: ACM manages network and host resources to support real-time connections. The services provided must be effective in the sense that both the quality-of-service provided to the individual connections and the utilization of system resources should be maximized.
- *Sensitivity to mission-specific requirements*: ACM must recognize the diverse requirements of the applications it supports. For a given application, its requirements change as its mode of operation changes. To provide services that are consistent with the demands of the application, mission-specific requirements should be properly propagated to the host and network resource managers and correctly taken into account in making decisions on connection admission and resource allocation.

We address the above issues in the design of ACM. ACM has been implemented in the newer version of our real-time toolkit, NetEx, over an ATM network. NetEx is a library of communication primitives that enables user applications to participate in delay guaranteed communications [DLSZ97, SLDZ97]. Our implementation is compatible with several standards and recommendations on QoS framework such as [ATM95, IT96].

1.3 Previous Work

The U.S. DoD has laid special emphasis on improving the responsiveness, security and reliability of communication services that play a critical role in current and future military operations [DISA94, ABIS96]. In this context, connection management for guaranteed performance (delay, jitter, etc) is a well-researched topic. The communication is based on a simplex fixed-route connection called real-time channel, variations of the type defined in [Fer90]. A real-time channel is essentially a virtual circuit with performance guarantees. Various connection management approaches for guaranteeing performance of real-time channels have been suggested in [Par92, AKRS94, KSF94, MZ95, Cru95, MIS96, Rah96, DLSZ97]. Connection management is addressed in terms of scheduling policies, traffic regulation methods, and analysis of delay and buffer constraints. The aforementioned studies deal with QoS specified as fixed-values. Dynamism in connection management is discussed in [PVZ93, PZF94]. More recently, an architectural framework for adaptive resource management is reported in [HWVC97, HTG97]. Our study appropriately supplements previous work in developing communications service for mission critical applications. We identify and solve the important issues in adaptive connection management.

2 Overview of Adaptive Connection Management

In this section, we present a schematic of our adaptive connection management (ACM). We first discuss connection QoS specification and classification. We then discuss the functional blocks of ACM and their inter-relation for adaptation.

2.1 Connections

This sub-section outlines connection QoS specification and classification.

2.1.1 Connection QoS Specification

The periodic model is traditionally used to specify the QoS of a connection. The parameters are specified as the ordered triplet, (C, P, D), where C is the message size in bits generated periodically every P seconds, and each message has a deadline D seconds. The traditional model however specifies parameters with fixed values. We extend this model to specify parameters over a range of minimum and maximum values. The j-th connection has its QoS_j given by,

$$QoS_j = [QoS_j^{worst}, QoS_j^{best}] \quad (1)$$

where

$$QoS_j^{worst} = (C_j^{min}, P_j^{max}, D_j^{max}) \quad (2)$$

$$\text{and, } QoS_j^{best} = (C_j^{max}, P_j^{min}, D_j^{min}) \quad (3)$$

If ACM admits a connection, the connection is offered an operating QoS, QoS_j^{op} such that,

$$QoS_j^{worst} \leq QoS_j^{op} \leq QoS_j^{best} \quad (4)$$

Figure 2 shows this feasible QoS region. The objective of ACM is to offer the best QoS possible to an admitted connection based on resource availability i.e., ACM keeps QoS_j^{op} as close to QoS_j^{best} as possible.

2.1.2 Connection Classification

ACM distinguishes connections to be from one of three classes: *critical*, *essential* and *non-essential*.

- *Critical* connections are of the highest criticality and are always admitted by ACM. Reserving resources a priori for critical connections ensures this. A critical connection, once admitted, cannot be preempted.
- *Essential* connections are of a criticality lower than critical connections but higher than the non-essential ones. An essential connection may be denied admission if sufficient resources are not available, but once admitted, it cannot be preempted.
- *Non-essential* connections are of the lowest criticality. They may be denied admission and be preempted in order to admit other connections of higher criticality.

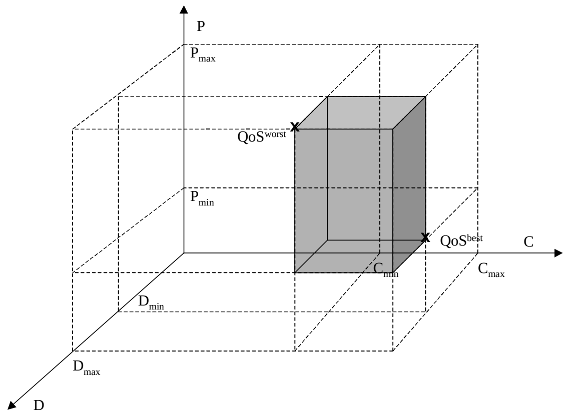


Figure 2. Feasible QoS Region

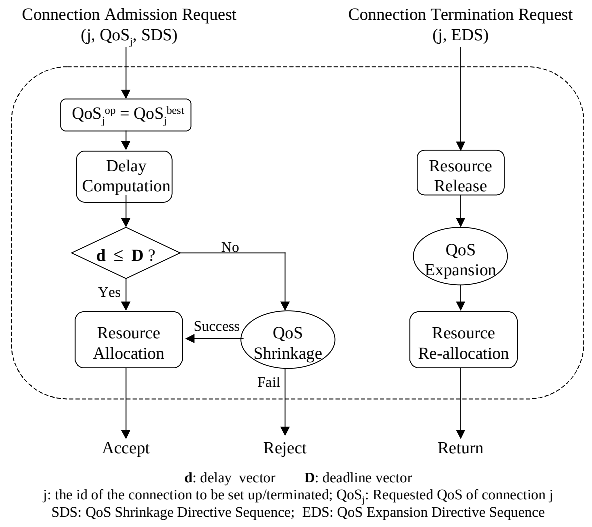
Such a criticality-based classification is appropriate for mission critical system where tasks inherently are of different criticality [MZB90, HWC97, IT96]. In the absence of this classification, a less critical task would use the same amount of resources as a more critical one. This clearly results in poor management of resources especially when there is resource contention. Further, classifying connections helps applications exploit the adaptation services provided by ACM resulting in better overall performance.

2.2 Adaptation Strategies

We separate adaptation strategies in ACM (See Figure 3) into two major threads - one for connection admission and another for connection termination. The ACM schematic builds on TCM (shown in Figure. 1). The main addition is a QoS Shrinkage module for connection admission and a QoS Expansion module for Connection Termination.

2.2.1 Adaptive Connection Admission

A Connection Admission Request (CAR) comes with the following parameters: $\{M_j, QoS_j, SDS_j\}$, where M_j is the j-th connection and QoS_j is the QoS for M_j as defined in Equation (1). The connection request specifies an adaptation policy via the Shrinkage Directive Sequence (SDS_j). SDS_j directs the ACM as to which connections' QoS need to be shrunk in order to admit the new connection.



Connection Admission Request
(j, QoS_j, SDS)
Connection Termination Request
(j, EDS)
$QoS_j^{op} = QoS_j^{best}$
Delay Computation
$\mathbf{d} \leq \mathbf{D}$?
No
Yes
Resource Allocation
Success
QoS Shrinkage
Fail
Resource Release
QoS Expansion
Resource Re-allocation
Accept
Reject
Return
d: delay vector **D**: deadline vector
j: the id of the connection to be set up/terminated; QoS_j: Requested QoS of connection j
SDS: QoS Shrinkage Directive Sequence; EDS: QoS Expansion Directive Sequence

Figure 3. Adaptive Connection Management

We now outline the different steps in adaptive connection admission.

C1: *Initial QoS Assignment.* For every incoming connection request, ACM sets the QoS_i^{op} of the new connection to QoS_i^{best} so that a new connection may be admitted at the best QoS requested if sufficient resources are available.

C2: *Delay Computation*. ACM then computes the upper bound on the delay **d** due to the new connection being admitted at QoS_i^{best} Delay computation is a very important part of ACM. This problem has been solved for ACM in [Li98].

C3: *Delay Test*. The next step is to tests if all the requested deadlines can be met i.e. $\mathbf{d} \leq \mathbf{D}$ (where **d** and **D** are respectively the vectors of delays and deadlines for a set of connections). The new connection is admitted only if the deadlines of this and the existing connections can be guaranteed.

C4: *QoS Shrinkage.* If the guarantee test fails, the ACM, based on SDS, proceeds to determine the level to which the QoS of a selected set of connections needs to be shrunk to successfully admit the new connection at QoS_j^{op}. If QoS Shrinkage is successful in admitting a connection by reducing QoS of some connections, resource allocation follows. A connection is rejected only when adaptation fails to free up enough resources to admit the new connection. This module is discussed in detail in Section 3.1.

C5: *Resource allocation.* Once a new connection is admitted, ACM allocates resources for the new connection such that the connection operates at QoS_j^{op}.

2.2.2 Adaptive Connection Termination

A Connection Termination Request comes with following parameters: $\{M_j, EDS_j\}$, where M_j is the j-th connection. The adaptation policy is specified by the Expansion Directive Sequence (EDS_j). EDS_j directs the ACM as to which connections are to get an increased QoS as result of resources released by the terminating connection.

T1: *Resource Release.* When a valid CTR arrives, ACM releases all resources that were reserved for the connection during admission time.

T2: *QoS Expansion*. ACM, based on EDS, then determines the level to which the QoS of a selected set of connections can be increased using the resources released at connection termination. This is described in detail in Section 3.2.

T3: *Resource Reallocation*. ACM then re-allocates resources needed to support the increased QoS.

3 Adaptation Modules

In the previous section we described an overview of ACM. As shown in Figure 3, the components of ACM at the core of the adaptation process are the *QoS Shrinkage* and *QoS Expansion* modules. These two modules, though independent of each other, work towards a common goal: providing better performance for ACM. QoS Shrinkage module helps ACM admit more number of connections by shrinking QoS of connections specified in the shrinkage directive sequence (SDS). QoS Expansion module, on the other hand, enables ACM to provide better offered QoS to existing connections by expanding QoS of candidate connections identified by the expansion directive sequence (EDS).

In this section we discuss *QoS Shrinkage* and *QoS Expansion* modules shown in Figure 3. We present formal definition of various terms used in these modules and then elucidate their operational details.

3.1 QoS Shrinkage Module

Recall from Figure 3 that a QoS shrinkage module is invoked by ACM when a new connection cannot be admitted with its QoS^{best}. ACM sends a *QoS Shrinkage Order* to this module to perform shrink operation. QoS Shrinkage Order carries the connection id of the new connection and a *Shrinkage Directive Sequence* (SDS) as input to this module. When the QoS shrinkage module finishes its execution of the QoS Shrinkage Order, it reports a success or failure to ACM. The decision of this module is then used by ACM to determine whether to accept or reject the new connection.

3.1.1 Shrinkage Directive Sequence

A Shrinkage Directive Sequence is an ordered list of shrinkage directives. Let a connection M_j come with a shrinkage directive sequence *SDS*. Let the *SDS* contain *n* shrinkage directives $SD_1, SD_2, \ldots, SD_n$. Then

$$SDS = \langle SD_1, SD_2, \ldots, SD_n \rangle. \qquad (5)$$

Further, a shrinkage directive $SD_l, 1 \le l \le n$ is defined as

$$SD_l = \{G_l, \boldsymbol{QoS}_{G_l}^{low}\}, \qquad (6)$$

where G_l is the set of connections that shrinking should be applied to, and $\boldsymbol{QoS}_{G_l}^{low}$

is the ordered list of QoSs for the connections in G_l i.e.

$$\boldsymbol{QoS}_{G_l}^{low} = \langle QoS_i^{low} \mid M_i \in G_l \rangle, \qquad (7)$$

where QoS_i^{low} of connection M_i is given by the ordered triplet (C_i^{low}, P_i^{low} D_i^{low}) and connection M_i may be shrunk down to QoS_i^{low} while trying to admit the connection M_j.

3.1.2 Shrinkage Operation

Having defined various terms in the previous section, we will now give a detailed description of the QoS shrinkage operation of ACM. For this we will refer to Figure 4 that shows the flow chart of the QoS shrinkage module.

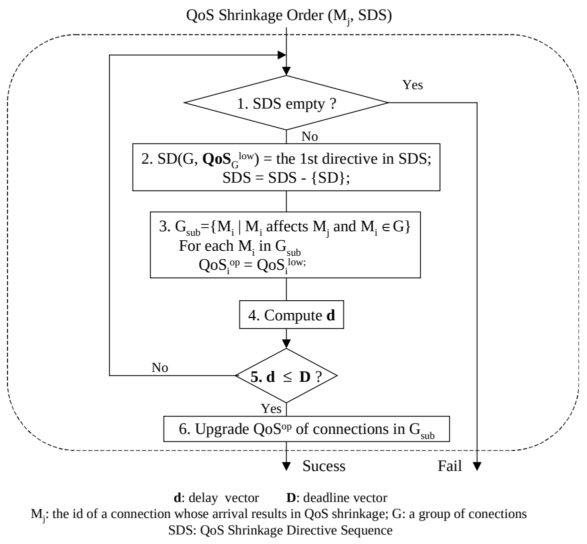

QoS Shrinkage Order (M_j, SDS)
1. SDS empty ?
Yes
No
2. SD(G, $\boldsymbol{QoS}_G^{low}$) = the 1st directive in SDS; SDS = SDS - {SD};
3. $G_{sub}=\{M_i \mid M_i$ affects M_j and $M_i \in G\}$ For each M_i in G_{sub} $QoS_i^{op} = QoS_i^{low}$;
4. Compute **d**
No
5. **d** ≤ **D** ?
Yes
6. Upgrade QoS^{op} of connections in G_{sub}
Sucess
Fail
d: delay vector
D: deadline vector
M_j: the id of a connection whose arrival results in QoS shrinkage; G: a group of conections
SDS: QoS Shrinkage Directive Sequence

Figure 4: QoS ShrinkageModule

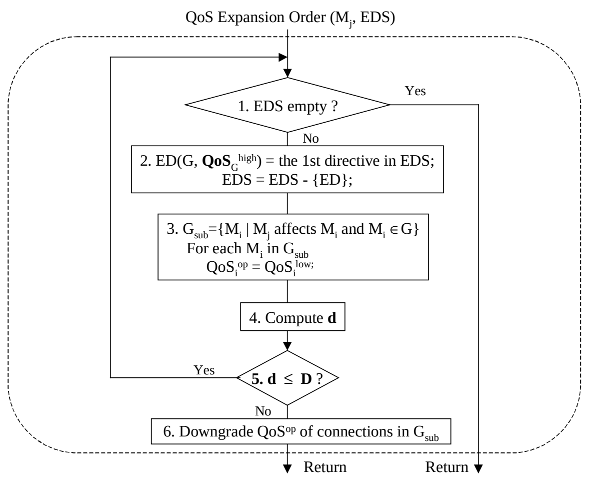
When a QoS Shrinkage Order arrives, the QoS shrink module does the following.

1) Checks if the shrinkage directive sequence is empty. If it is, then it reports a failure.
2) Dequeues the first shrinkage directive from the shrinkage directive sequence and gets the group of connections G in the shrinkage directive.
3) Identifies a subgroup G_{sub} of G such that G_{sub} consists of only those connections in the group G which affect the connection M_j. This makes sure that ACM does not alter operating QoS of connections that do not share resources with the connection M_j.
4) Computes delays of all the connections in G_{sub}.
5) Checks if delays of all the connections in G_{sub} are less than their respective deadlines (**d** ≤ **D**). If the condition is false, then enough resources are not available to admit the new connection M_j. Control then passes to step 1.
6) Otherwise, it tries to upgrade the operating QoS of all the connections in G_{sub}. When it comes out of the iteration, it would have shrunk operating QoS of all the connections in G_{sub} to their respective QoS^{low}. But there may be some resources available that can upgrade operating QoS of these connections to a value higher than QoS^{low}. This is achieved by performing binary search between QoS_l^{low}

and QoS_l^{best} of all connection $M_l \in G_{sub}$ while the delay test is still satisfied. This ensures that ACM gives as high a QoS as possible to the connections in G_{sub} with the available resources. A *success* is then reported.

Preemption of a connection M_l can be specified by setting QoS_l^{low} to be zero i.e. QoS_l^{low} should be $(0, \infty, \infty)$. Critical and essential connections cannot be preempted, hence for a critical or essential connection M_l, $QoS_l^{low} \geq QoS_l^{worst}$. Non-essential connections, on the other hand, can be preempted, so $QoS_l^{low} \geq 0$ for such class of connections. The QoS shrinkage module can also be invoked when the user sends absolute shrinkage directives i.e. there is no new connection that needs to be admitted, but the user wants to shrink operating QoSs of some of the existing connections. The flowchart for the absolute shrinkage directive is very similar and is not shown in this paper.

3.2 QoS Expansion Module



QoS Expansion Order (M_j, EDS)
1. EDS empty ?
Yes
No
2. ED(G, QoS_G^high) = the 1st directive in EDS; EDS = EDS - {ED};
3. G_sub={M_i | M_j affects M_i and M_i ∈ G} For each M_i in G_sub QoS_i^op = QoS_i^low;
4. Compute d
5. d ≤ D ?
Yes
No
6. Downgrade QoS^op of connections in G_sub
Return
Return

d: delay vector **D**: deadline vector
M_j: the id of a connection whose termination results in QoS expansion; G: a group of conections
SDS: QoS Expansion Directive Sequence

Figure 5: QoS Expansion Module

Figure 3 shows that QoS expansion module is invoked by ACM after it releases resources of connection M_j subsequent to a connection termination request for the connection. ACM sends a *QoS Expansion Order* to this module to perform expansion operation. QoS Expansion Order carries the connection id of the connection to be torn down and an *Expansion Directive Sequence* (EDS) as input to this module.

The expansion operation is very symmetric to the shrinkage operation. Hence operation of QoS expansion module and some definitions will very closely resemble those of QoS shrinkage module. Although repetitious, we will present them for the sake of completeness.

3.2.1 Expansion Directive Sequence

Expansion Directive Sequence is an ordered list of expansion directives. Let a connection M_j come with a expansion directive sequence *EDS*. Let the *EDS* contain *n* expansion directives $ED_1, ED_2, \ldots, ED_n$. Then

$$EDS = \langle ED_1, ED_2, \ldots, ED_n \rangle. \tag{8}$$

And expansion directive ED_l, $1 \leq l \leq n$ is defined as

$$ED_l = \{G_l, \boldsymbol{QoS}_{G_l}^{high}\}, \tag{9}$$

where G_l is the set of connections to participate in the expansion process, and $\boldsymbol{QoS}_{G_l}^{high}$ is the ordered list of QoSs of connections in G_l i.e.

$$\boldsymbol{QoS}_{G_l}^{high} = \langle QoS_i^{high} \mid M_i \in G_l \rangle, \tag{10}$$

where QoS_i^{high} of connection M_i is given by the ordered triplet $(C_i^{high}, P_i^{high}\ D_i^{high})$ and connection M_i may be expanded to QoS_i^{high} during expansion operation.

3.2.2 Expansion Operation

In order to elucidate the expansion operation, we will refer to Figure 5. When a QoS expansion order arrives, the expansion module does the following.

1) Checks if the expansion directive sequence is empty, in which case it returns.
2) Dequeues the first expansion directive ED from the expansion directive sequence and gets the group of connection G in the expansion directive.
3) Identifies a subgroup G_{sub} of G such that G_{sub} consists of only those connections in the group G that are affected by the connection M_j. If this step is not followed, then QoS of some connections may be expanded even though they do not share resources with M_j. This may increase delay of some connections that are not in G. Since delay test is confined to G, this may cause some of these connections (that are not in G) to miss their deadlines.
4) Computes delays of all the connections in G_{sub}.
5) Checks if delays of all the connections in G_{sub} are less than their respective deadlines (**d** ≤ **D**). If the condition is true, then it means that there are more resources available in the system and further expansion is possible. So it goes back to step 1.
6) Otherwise, tries to downgrade operating QoS of all the connections in G_{sub} and then returns. This is achieved by performing binary search between QoS_l^{high} and QoS_l^{worst} of all connection $M_l \in G_{sub}$ and ensuring that the delay test is not violated. This operation ensures as high QoS as possible with the available resources for the connections in G_{sub}.

For a connection M_j belonging to any class (critical, essential or non-essential), $QoS_i^{high} \leq QoS_i^{best}$. Thus a connection never operates at QoS higher than the best QoS. Resumption of a connection can also be achieved by specifying appropriate QoS_i^{high}. Similar to the QoS shrinkage module, this module can also be invoked by absolute expansion order.

4 Implementation and Evaluation

In this section we discuss the implementation and performance evaluation of adaptive connection management.

4.1 Implementation in NetEx

ACM has been implemented in the newer version of our real-time toolkit, NetEx [SLDZ97]. NetEx is a communication software package developed in the Department of Computer Science at Texas A&M University. NetEx is a library of communication primitives that enables user applications to participate in delay guaranteed communications. NetEx consists of three main components: user library, Host Traffic Manager (HTM) and Network Traffic Manager (NTM). User library is the interface of NetEx to the end users. HTM is the module responsible for managing and policing traffic at the host. NTM is primarily responsible for connection management of the entire system. For a detailed description of NetEx architecture please refer to [SLDZ97].

The design of a software architecture like NetEx that provides delay guarantees to adaptive real-time connections is influenced by many potentially conflicting requirements. In the previous version of NetEx, we used TCM for connection management. In the new version we have replaced TCM with ACM described in this paper. We have used this new version of NetEx to evaluate the performance of our ACM. While supporting adaptive connection management, NetEx is designed to be compatible with existing operating systems and network technologies. Some other solutions call for changes to the OS kernel or network protocols, which may not be cost-effective. NetEx strives to provide delay guarantees on existing OS platforms and networks. Such compatibility makes it easy for NetEx to support many legacy applications. Furthermore, the technology we develop accommodates the evolution of existing platforms with the addition of new devices. NetEx can thus continue to be effective and efficient while the underlying systems grow in size and extend to heterogeneous domains.

4.2 Experimental Setup

For our experiments, we have three Sun Sparc 4 workstations running Solaris 2.5 connected to an ATM switch (Fore Systems' ForeRunner) with link speed of 155 Mbps. One of the workstations runs as an NTM host (with the new ACM) and the other two are run as HTM hosts. User applications run on these two HTM hosts. An application on one HTM host generates a request to connect to another application on the other HTM host. Connection requests have Poisson arrivals. All admitted connections have an exponentially distributed lifetime. QoS is specified as a range with (C^{min}, P^{max}, D^{max}) as the lower limit (QoS^{worst}) and (C^{max}, P^{min}, D^{min}) as the upper limit (QoS^{best}). Average C is generated from a uniform distribution and C^{min} is set to 5% lower and C^{max} is set to 5% higher than this average value. P^{min} and P^{max} are calculated according to the required utilization of the system. Deadline ranges are made equal to the corresponding periods i.e. $D^{min}= P^{min}$ and $D^{max}= P^{max}$. 10% of the connections generated are critical, 30% are essential and the rest are non-essential connections. When a connection is accepted, its operating QoS QoS^{op} is given by (C^{op}, P^{op}, D^{op}).

We used two configurations of Adaptive Connection Management:

- *Configuration S*: In this configuration, only QoS shrinkage is allowed, QoS expansion module is turned off. The shrinkage directive sequences are such that the QoS of connections are shrunk in the increasing order of their criticality i.e. non-essential connections are shrunk first, followed by essential and then critical. Preemption of any connection is not allowed.
- *Configuration S&E*: This is same as the Configuration S except that QoS expansion is also allowed when a connection leaves the system. The Expansion directive sequences are such that the QoS of connections are expanded in the decreasing order of their criticality i.e. critical connections are expanded first, followed by essential and non-essential connections.

To compare performance of ACM with TCM we also set up the following two baseline configurations.

- *Configuration B*: In this configuration, no adaptation is performed and connections are admitted based on their QoS^{best}. This configuration corresponds to TCM with applications sending connection admission request with a single-valued QoS that corresponds to QoS^{best} of ACM.
- *Configuration W*: This configuration is the same as Configuration B except that connections are admitted based on their QoS^{worst}.

4.3 Performance Metrics

We will use the following performance metrics to evaluate our ACM :

- *Admission Probability (AP)*: Admission Probability of a connection management system indicates the likelihood of a connection being admitted. We define admission probability as

$$AP = \frac{Number\ of\ connections\ admitted}{Total\ number\ of\ connections\ requested\ for\ admission}$$

- *QoS Effectiveness (QOSE)*: The QoS effectiveness of a connection is a measure of how close the operating QoS of the connection is to the maximum QoS asked by the connection. It is defined as

$$QOSE = \frac{\sqrt{(C^{op} - C^{\min})^2 + (P^{op} - P^{\max})^2 + (D^{op} - D^{\max})^2}}{\sqrt{(C^{\max} - C^{\min})^2 + (P^{\min} - P^{\max})^2 + (D^{\min} - D^{\max})^2}}$$

- *Average Execution Time (AET)*: Execution time is the amount of time an application has to wait to get a reply from ACM after submitting a connection admission request. Average Execution Time of *N* connections is defined as

$$AET = \frac{Sum\ of\ execution\ time\ of\ N\ connections}{N}$$

4.4 Experimental Results

Figures 6, 7, and 8 show variation of AP, QOSE and AET respectively as utilization of the system increases from 0% to 100%.

We make the following observations from these figures:

- For all configurations, AP decreases as utilization increases. As system utilization increases, the availability of system resources decreases. Hence, more connections get rejected.
- AP for Configuration S, S&E and W are same for all utilization values. This is so because a connection is always admitted with QoS^{worst} in Configuration W and the same connection can also be admitted in S and S&E with $QoS^{op} \geq QoS^{worst}$. But AP for Configuration B is much lower than the other three. In this case, connections are admitted with QoS^{best}. Hence the system allocates more resources to each admitted connection than it does in other configurations. As a result, the AP is lower than the other three configurations.
- QOSE for Configuration B is always 1.0 and that for Configuration W is always 0.0. This is what is expected because ACM admits connections with operating QoS equal to QoS^{best} and QoS^{worst} respectively in those two configurations.

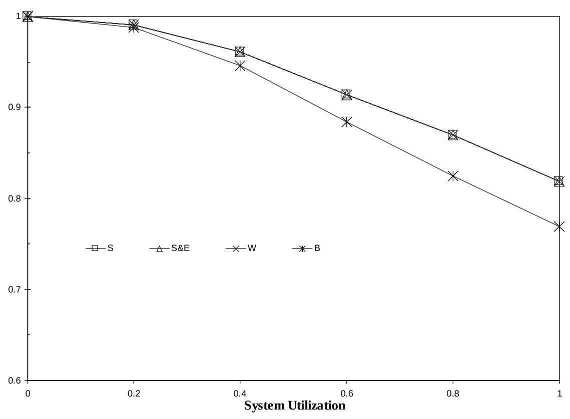


Figure 6: Admission Probability vs System Utilization

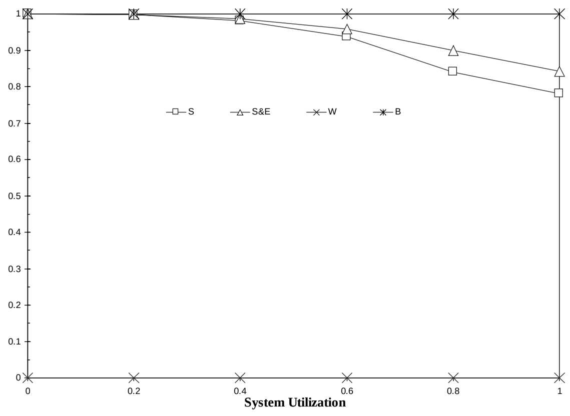


Figure 7: QoS Effectiveness vs System Utilization

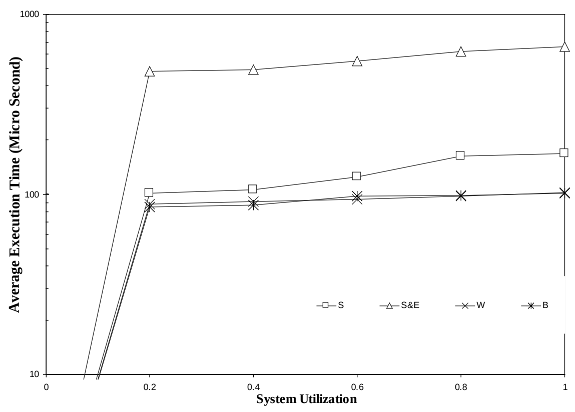


Figure 8: Average Execution Time vs System Utilization

- QOSE for Configuration S and S&E are lower than that for Configuration B. This is due to the fact that at higher utilization, for Configuration S and S&E, the shrinkage module has to shrink the QoS of a larger number of existing connections to admit a new connection. This brings the QOSE down. Although Configuration B performs better than S and S&E in terms of QOSE, recall that the AP for Configuration B is much lower than both S and S&E.
- QOSE of Configuration S&E is always higher than that for Configuration S. In Configuration S&E, ACM expands operating QoS of existing connections, when a connection leaves, which boosts its QOSE. This expansion operation does not happen in Configuration S.
- For all the configurations, AET increases as system utilization increases. This is because at higher utilization, the system has more active connections to deal with which makes the execution time longer. AET for Configuration SE is the highest, since in this case ACM does the largest amount of work (in terms of shrinkage and expansion operation). The overall AET of Configuration S&E is very low (less than 1 millisecond), which is very good for systems in practice.

5 Conclusion and Future Work

In this paper we have introduced adaptive connection management that addresses the shortcomings in traditional connection management. We have demonstrated that by taking an adaptive approach to connection management, we can enhance communication support for mission critical applications.

The main contributions of this paper are:

1. An enhanced QoS model: We have extended the traditional fixed-value QoS model to one that accepts QoS specified over a range. This gives the applications and the management flexibility in resource allocation.
2. QoS support for connection classes: This is particularly suitable to mission critical applications that need criticality-based connection management.
3. Efficient and effective QoS adaptation modules: With QoS Shrinkage and QoS Expansion modules, the resources are dynamically re-allocated in order to meet the need of mission critical real time connections. Our performance data shows that the additional cost (in terms of execution time) of adaptation is low while the benefits are high.
4. Specially designed adaptation directives: By this mechanism, mission-specific requirements can be properly propagated to our adaptive connection management system and correctly taken into account in making decisions on connection admission and resource allocation.
5. Connection-level flexibility: ACM provides dynamic connection-level flexibility via user supplied directives. A connection request can carry directives to shrink or expand the QoS of existing connections in favor of the incoming connection. This connection-level flexibility enhances services to be consistent with the demands of mission critical applications.
6. A successful implementation in NetEx: The proposed adaptive connection management scheme has been implemented in NetEx. NetEx is a toolkit of communication primitives for delay-guaranteed communications. Data collected from our experiments with the implementation confirms our thesis that the overall performance of the system is improved when connection management responds to dynamic fluctuations in resource availability.
7. Practical and compatible technology: Our proposed scheme is compatible with existing network standards and industrial practices. NetEx is realized with network

products which are currently commercially available and does not require any changes to them.

Several extensions to adaptive connection management are possible. We are currently adding fault-tolerant techniques so that the overall system will adaptively meet the real-time and fault-tolerance requirements of mission critical applications.

Acknowledgements

This work was partially sponsored by AFOSR, AFMC, USAF under a grant (number F49620-96-1-1076) and by Texas Higher Education Coordinating Board under its Advanced Technology Program and by the DARPA through Honeywell Technology Center under a contract (number B09333438).

6 References

[ABIS96] ABIS Task Force, DoD Advanced Battlespace Information Systems Task Force Report, 1996.

[AKRS94] C. M. Aras, J. F. Kurose, D. S. Reeves, and H. Schulzrinne, "Real-time communication in packet-switched networks," Proceedings of the IEEE, pp. 122-139, Jan. 1994.

[ATM95] ATM Forum, ATM UNI Specification Ver. 3.1, 1995.

[BKHDP96] G. v. Bochman, B. Kerherve, A. Hafid, P. Dini, A. Pons, "Architectural Design of Adaptive Distributed Multimedia Systems," IEEE International Workshop on Multimedia Software Development, 1996, pp. 31-40

[Cru95] R. L. Cruz, "Quality of Service Guarantees in Virtual Circuit Switched Networks," IEEE Journal of Selected Areas in Communication, special issue on "Advances in the Fundamentals of Networking" (vol. 13 no. 6 pp. 1048-1056), August 1995.

[DISA94] Defense Information Systems Agency, Baseline Common Operating Environment, 1994.

[DLSZ97] B. Devalla, C. Li, A. Sahoo, W. Zhao, "Connection-Oriented Real-Time Communication for Mission Critical Applications: An Introduction to NetEx," Proceedings of NAECON '97, pp. 698-707, July 1997.

[Fer90] D. Ferrari, "Client requirements for real-time communication services," IEEE Communications Magazine, vol. 28, pp. 65-72, Nov. 1990.

[HTG97] J. Huang, R. Jha, W. Heimerdinger, M. Muhammad, S. Lauzac, B. Kannikeswaran, K. Schwan, W. Zhao, R. Bettati, " RT-ARM: A Real-Time Adaptive Resource Management System for Distributed Mission-Critical Applications," IEEE Workshop on Middleware for Distributed Real-Time Systems & Services '97

[HWC97] J. Huang, Y. Wang, F. Cao, "On Developing Distributed Middleware Services for QoS- and Criticality-Based Resource Negotiation and Adaptation," Honeywell Technical Report SST-P97-012, June 1997.

[HWVC97] J. Huang, Y. Wang, N.R. Vaidyanathan, F. Cao, "GRMS: A Global Resource Management System for distributed QoS and Criticality Support," Proc. of COMPSAC, 1997.

[IT96] "Information Technology- Quality of Service Framework," DIS 13236, ISO/IEC JTC1/SC21 N10339, June 1996.

[KSF94] D. D. Kandlur, K. G. Shin, and D. Ferrari, "Real-time communication in multi-hop networks," IEEE Transactions on Parallel and Distributed Systems, pp. 1044-1056, Oct. 1994.

[Li98] C. Li, "Design of Adaptive Connection Admission Control for Hard Real-time ATM Networks," M.S. Thesis, Texas A&M University, May 1998.

[MIS96] A. Mehra, A. Indiresan, and K. Shin, "Resource management for real-time communication: Making theory meet practice." Proceedings of RTAS, 1996.

[MZ95] N. Malcolm and W. Zhao, "Hard Real-Time Communication in Multiple- Access Networks," Journal of Real-Time Systems, January 1995.

[MZB90] N. Malcolm, W. Zhao, C. Barter, "Guarantee Protocols for Communication in Distributed Hard Real-Time Systems," Proc. of IEEE INFOCOM, 1990, pp. 1078-1086.

[Par92] A. K. J. Parekh, A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1992.

[PVZ93] C.J. Parris, G. Ventre, H. Zhang, "Graceful Adaptation of Guaranteed Performance Service Connections", Proc. of GlobeCom 1993.

[PZF94] C.J. Parris, H. Zhang, D. Ferrari, "A Dynamic Management Scheme for Real-Time Connections," Proc. IEEE INFOCOM '94, Toronto, Ontario, Canada 1994, pp. 698-707.

[Rah96] A. Raha. *Real-Time Communication in ATM Networks.* Ph.D. Thesis. Texas A&M University, 1996.

[RKZ95] A. Raha, S. Kamat, W. Zhao, "Guaranteeing End-to-End Deadlines in ATM Networks", Proc. of International Conference on Distributed Computing System 1995.

[SLDZ97] A. Sahoo, C. Li, B Devalla, W. Zhao, "Design and Implementation of NetEx: A Toolkit for Delay Guaranteed Communications," Proceedings of MilCom '97.

7 Author Biographies

Anirudha Sahoo holds a B.S. in Electrical Engineering from R.E.C., Rourkela, India, and an M.S. in Computer Science from Univ. of SW Louisiana, Lafayette, LA. He is currently a Ph.D. student in Computer Science at Texas A&M. His research interests are ATM-based heterogeneous networks, Virtual LANs and LAN switching.

Badari Devalla received B.E. in Electronics and Communication Engineering from Anna University, Madras, India in 1991 and M.S. in Electrical Engineering from Texas A&M University, College Station, TX in 1994. He is currently a Ph.D. student in Computer Science at Texas A&M. His primary research interests are Quality of Service guarantees in high-speed networks.

Yong Guan received B.S. and M.S. in Computer Science in 1990 and 1996, from Peking University, China. He is currently a Ph.D. student in the Computer Science department at Texas A&M. His research interests are fault-tolerant real-time communications and gigabit switched networks.

Dr. Riccardo Bettati received his Diploma in Computer Science from the Swiss Federal Institute of Technology, Zurich in 1988 and Ph.D. in Computer Science from the University of Illinois at Urbana-Champaign in 1994. After two years as of post-doctoral research at the International Computer Science Institute and the University of California, Berkeley, he joined the Department of Computer Science at Texas A&M where he is currently an assistant professor. His research interests are in real-time computing and communications, large-scale distributed systems, and middleware for distributed real-time computing.

Dr. Wei Zhao received his B.Sc. in physics from Shaanxi Normal University, Xian, China, M.Sc. and Ph.D in computer and information science from the University of Massachussetts, Amherst, MA, in 1983 and 1986, respectively. In 1990 he joined the Department of Computer Science at Texas A&M University where he is currently a full professor and the department head. His current research interests include real-time computing and communication, distributed operating systems, database systems and fault-tolerant systems.