Design and Implementation of NetEx: A Toolkit for Delay Guaranteed Communications

A. Sahoo, C. Li, B. Devalla, and Wei Zhao
Department of Computer Science, Texas A&M University
College Station, TX 77843
email: {asahoo, cenli, badari, zhao}@cs.tamu.edu

ABSTRACT

In this paper, we report the design and implementation of a software suite called NetEx (short for Network Express) that can provide connection-oriented delay-guaranteed communication services at application level. With NetEx, user applications request connection set-up by specifying their traffic (using some standard traffic descriptors) and quality of service (e.g., delay bound). NetEx runs a connection admission control (CAC) algorithm to check if the new connection can be accepted. NetEx integrates network wide admission control with host run-time traffic control to facilitate application-to-application delay-guaranteed communication. The modular design and implementation of NetEx allows it to also serve as a software test-bed to experiment different network management techniques.

1. Introduction

In this paper, we report our project on providing connection-oriented delay-guaranteed communication services at application level for distributed mission critical systems. Examples of these systems include industrial process control, space program, and military command and control systems.

Much of the existing work in real-time communications has concentrated on providing deadline guarantees at the MAC (Media Access Control) level. We propose to provide connection-oriented and delay-guaranteed communication services for distributed mission critical systems at the application level. To achieve our objective, various network and host resource management techniques have been developed [5, 11-13]. These techniques are now realized in a software tool kit, *NetEx*. We describe the design and implementation of NetEx and address the fundamental technical issues associated with it.

NetEx comprises of Host Traffic Manager (HTM) and Network Traffic Manager (NTM). HTM resides in each participating host while NTM can be implemented either at a central host or in a distributed manner. HTM performs traffic scheduling and enforces traffic regulation at individual hosts. NTM is the decision maker on connection admission control (CAC). With NetEx, user applications request connection set-up by specifying their traffic (using some standard traffic constraint functions) and quality of service (e.g., delay bounds). NetEx runs a connection admission control procedure to check if the new connection can be accepted. This admissibility ensures that the deadline requirements of the new and existing connections can be guaranteed.

The following features highlight innovations in the design of NetEx:

- *Integrated and consistent control strategy:* A coordinated effort by different system components enables NetEx to provide delay guarantees at the application level. Connection admission has to be controlled so that only the connections whose admission will not violate any deadline and resource constraints will be accepted. At run time, traffic of individual connections should be monitored and regulated (at the host) to prevent any over-use of resources. NetEx is designed to address these management and control issues in an integrated and consistent manner, details of which are given in Section 2.
- *An efficient methodology for deriving the worst case delays:* In order to guarantee the worst case message delays, the system must be able to predict them accurately. To achieve this, NetEx resorts to a decomposition approach for delay analysis. Details about this method is described in Sections 3 and 4.
- *Extensibility:* The modular design and implementation of NetEx allows it to also serve as a software test-bed to experiment different connection admission and traffic management techniques. Enhancements in terms of fault tolerance support and security are also possible.

Our work complements previous work on providing delay guaranteed communication services. Guaranteeing delay bounds in both multiple access and point-to-point networks has been extensively studied [1-3]. The solutions typically provide MAC-to-MAC guarantees. For an inter-network environment, the generic solution consists of connection-oriented communications with some form of admission control and traffic regulation (typically based on packet scheduling at the network interface) [4-7]. From an implementation standpoint, several protocols have also been proposed for guaranteed services on networks. Resource Reservation Protocol (RSVP) [8] is a receiver-oriented set-up protocol for connectionless networks for providing Quality of Service (QoS) guarantees. ATM Forum's UNI

Signaling protocol (derived from ITU-T Q.2931) describes QoS and traffic parameters specification procedures [9].

2. Overview of NetEx Traffic Control Strategy

In this section, we will first discuss two important control components involved in providing delay-guaranteed services at the application level. These components are system connection admission control (CAC) and host run-time traffic control (RTTC). We then briefly discuss the design strategy of these control mechanisms in NetEx.

2.1 System Connection Admission Control

As mentioned before, NetEx provides connection-oriented services at the application level. That is, before communication between two applications (resident on different hosts) ensues, some involved application (called the *initiator*) must request NetEx to establish a connection. An *initiator* can be the sender, receiver or a third application. In NetEx, the system connection admission control module (CAC) is responsible to decide if the requested connection can be established and if so, to allocate proper amount of resource to it. Following steps are carried out in the CAC:

1. *Validation of the CAC request*. This is to check if the initiator, the sender, and the receiver applications have proper rights for establishing the requested connection.

2. *Route generation*. Selection of a route for the requested connection can be made by using different criteria. For the purpose of efficiency, in the current version of NetEx, a route with the shortest path is used. Our performance study has found that this method performs reasonably well in comparison with others while having a minimum run-time overhead [10].

3. *Delay derivation and testing*. This is the most critical step. We first derive the delay bound of the new connection and then test if it is no more than the deadline requested. Since the introduction of the new connection may impact the delays of some existing connections, their delays will also have to be re-derived and tested. In any case, the application-to-application worst case delay, d, of a connection can be expressed as

$$d = d_{sender} + d_{network} + d_{receiver} \quad (1)$$

where d_{sender}, $d_{network}$, and $d_{receiver}$ are the worst case delays suffered at the source host, the network, and the destination host, respectively. We will discuss our methodology on network and host delay analyses in Sections 3 and 4, respectively.

2.2 Run Time Traffic Control

The system wide connection admission control alone cannot guarantee message delays. Run time traffic control in every host also plays an important role towards achieving this goal. In NetEx, the run time traffic control module is responsible for the following functions:

1. *Monitor connection traffic*. NetEx does not allow a misbehaving connection to worsen performance of others. A connection that violates its specified traffic constraints will be terminated and an error message will be sent to its initiator.

2. *Regulate and schedule traffic of a connection from the sender host to the network.* By doing this, NetEx avoids two problems that may potentially degrade system performance:

- *Priority inversion:* A message with a smaller deadline may be blocked by another message with a relative larger deadline, hence causing the former to miss its deadline.
- *Bursty traffic:* The peak rate of traffic from an application can be more than the average rate. If these peaks are not smoothened out, then they will cause considerable amount of burst in the network, consequently worsening the performance of other connections.

In NetEx, the above functions are realized with a *table driven* method. That is, the RTTC schedules packet transmissions based on an execution table. The execution table on a host consists of a number of execution slots where one or more packets can be transmitted per slot. The number of slots assigned to an individual connection is determined by the NTM at the connection admission time, based on the delay requirements. The RTTC continuously executes the table in a cyclic manner.

NetEx thus provides traffic monitoring, regulation, and scheduling via this table-driven mechanism. Messages of individual connections are stored in the buffer before RTTC transmits them. The buffer size is allocated consistently with the connection traffic rate and transmission schedules defined by the RTTC execution table. Thus, by monitoring the usage of the buffer, RTTC can indirectly detect misbehaving connections, those that have violated their traffic specification agreed upon at connection establishment time. The traffic regulation is achieved by placing the slots assigned to a connection appropriately in the table. This requires minimal support from the applications and operating system. This is an advantage especially in an environment where the OS does not provide real-time scheduling capability.

2.3 Integrated and Consistent Control Strategy

We would like to stress that in NetEx, connection admission control and run-time traffic control are carried out in an integrated and consistent manner. The integration and consistency are reflected as explained below:

1. Delay derivation is the most important task in CAC. Its success relies on correct modeling of the traffic in both network and hosts. Host traffic modeling is especially

challenging given the variety of activities in the hosts and their implementation strategies. By taking into account the effect of run-time traffic monitoring, regulation, and scheduling, our CAC is able to predict the traffic behavior of connections and effectively derive the delay bounds in the hosts. Details on this will be discussed in Section 4.

2. On the other hand, run-time traffic control is dictated by the CAC. The CAC balances the requirements of individual connections as well as the loading status of hosts and networks to decide how to regulate and schedule the traffic of connections. It is the CAC that allocates table slots for a connection and the RTTC faithfully follows this allocation to regulate traffic at the hosts. Thus the run-time traffic control works consistently with the objective of the CAC.

Hence, either admission control or run-time traffic control alone is insufficient to provide application level delay guarantee. They must be integrated and be consistent in order to obtain predictable and tight delay bounds for the messages transmitted in the system.

3. Network Delay Analysis

A key technique we develop with NetEx is to derive the delay bounds of networks and hosts so that (by using (1)) the CAC can make an admission decision based on application-to-application delay. We present our delay analysis techniques in this and next section.

3.1 Basic Approach

We use a decomposition approach to derive the delay bound of a cell in an ATM network. With this method, the ATM network is decomposed into a set of servers. A server can be a link, or a component of an ATM switch [5, 11]. Delays of ATM cells at each server are analyzed. The network delay of a cell from a particular connection is then the summation of the delays of servers the connection passes through. Formally, let G_j be the set of indices of servers which connection j passes through. Then, the worst case network delay for connection j is given by

$$d_{network} = \sum_{i \in G_j} d_{i,j} \qquad (2)$$

where $d_{i,j}$ is the worst case delay suffered by cells of connection j at server i.

This decomposition approach provides the basis for a general and modular delay analysis. Once the system is decomposed, *traffic description* and *server analysis* must be addressed to derive the end-to-end delay bounds. Traffic description is a parametric way of specifying the traffic characteristics of a connection. Given a server and the traffic description of the connection at the input, server analysis computes the worst case delay suffered by a connection passing through the server and the traffic at the output. We discuss these two issues in the next two subsections.

3.2 Traffic Description Function

Much of previous work described traffic at the source. But this may not be sufficient when the traffic pattern changes due to multiplexing within the network. In traditional networks, stochastic models have been used to describe traffic. They can evaluate average performance of networks. But they are not adequate for hard real-time systems, which require worst case analysis. For hard real real-time system, the maximum rate function, $\Gamma(I)$, has been used successfully [5]. We have also adopted the maximum rate function to describe traffic internally in NetEx. The maximum rate function is defined as the maximum data arrival rate in a time interval of length *I* units. That is, for connection *j*, its traffic at the entrance of server *i* is given by

$$\Gamma_{i,j}(I) = \max_{\forall t > 0} \left(\frac{\textit{number of bits arrived in the } \text{interval } (t, t+I]}{I} \right) \quad (3)$$

Use of $\Gamma(I)$ may be unsuitable due to following reasons: 1) It takes a large amount of memory space to store $\Gamma(I)$ if a closed form for $\Gamma(I)$ cannot be found; 2) It may not be feasible to extract the enormous amount of information needed to construct $\Gamma(I)$ for real world traffic. Approximation methods have been proposed to avoid these limitations [12]. In particular, the *point approximation method* is used to construct an approximate $\Gamma(I)$ when traffic information at limited number of points are given. Our experiments have shown that a six-point approximation can characterize real traffic with good precision [12]. NetEx uses the six-point approximation method to represent Γ(I) functions.

3.3 Server Analysis

The objectives of the server analysis are to obtain 1) the worst case delay suffered by a connection's cell at a server; 2) the buffer space needed at a server; and 3) the traffic description of a connection at the output of a server. Obviously, the first two objectives are directly related to the goal of network delay analysis. The traffic description at the output of a server is necessary in order to carry out analysis at the subsequent servers.

There are two kinds of servers in a network: 1) constant servers (e.g., transmission lines) and 2) variable servers (e.g., traffic regulators and multiplexers). Analysis of a constant servers is trivial and hence will not be discussed here. For variable servers, we will only demonstrate how to analyze a multiplexer with FIFO scheduling policy. Analysis of other variable servers and scheduling policies are not given here due to space limitation. Interested readers may refer to [5].

Let server i be a multiplexer with FIFO scheduling policy. Let Ω_i be the set of indices of connections that pass through server i. Let $\Gamma_{i,j}$ and $\Gamma'_{i,j}$ be the traffic description functions of connection j at the input and output of server i, respectively, where $j \in \Omega_i$. Using these notations, the maximum buffer requirement at server i, Q_i, is given by

$$Q_i = \max_I \left(\sum_{j \in \Omega_i} I \cdot \Gamma_{i,j}(I) - I \cdot p_i \right) \qquad (4)$$

where p_i is the transmission speed (bits/second) of server i. Let the real size of buffer at server i be S_i. $Q_i \leq S_i$ is a constraint that must be satisfied during connection admission control to avoid buffer overflow.

Furthermore, for a FIFO server, the cell whose arrival results in the maximum queue length, should suffer the worst case delay. Hence,

$$d_{i,j} = Q_i \, p_i \qquad (5)$$

For the description of output traffic at server i, we have the following results:

$$\Gamma'_{i,j}(I) = (I+d_{i,j})\, \Gamma_{i,j}(I + d_{i,j})/I \qquad (6)$$

That is, we can express the output traffic, $\Gamma'_{i,j}(I)$, in terms of the input traffic, $\Gamma_{i,j}(I)$ and the delay, $d_{i,j}$. (6) follows directly from the delay property of maximum rate function [1]. Using (6), subsequent servers can be analyzed. Once all the delays at individual servers are obtained, we can substitute (5) into (2) to compute the network delay for connection j.

4. Host Delay Analysis

From (1), analyzing delays at the sender and the receiver hosts is necessary to obtain the application-to-application delay. Recall that in NetEx, the host run-time traffic control (RTTC) uses a table driven (TD) mechanism to achieve a bounded delay in the host part of the connection. In this section, we will briefly explain the basic principle behind the RTTC table driven method and then derive the delay in the host.

4.1 Operations and Issues in RTTC

The RTTC schedules packet transmissions in an execution table. An execution table in a host consists of a number of execution slots. Each slot has fields specifying the connection identifier of the connection to be served in the slot, number of packets to be served and type of service (whether a send or a receive). Each slot corresponds to a fixed duration of time (T_{slot}). The RTTC wakes up at every T_{slot} time boundary and checks the corresponding slot to identify the connection(s) to be served and sends (or receives) the number of packets specified in the slot entry. Actual value of T_{slot} is dependent on the resolution of the timer provided by the underlying operating system (OS). The table is executed in a cyclic manner. The entries in the table are provided by the CAC when a request for connection establishment is granted. An auxiliary buffer called application buffer is used to hold all the packets that have not yet been dispatched by the RTTC. The RTTC runs in the highest priority mode that the local OS can provide. This guarantees that connections which should be served in a certain time slot will receive services without interference from other host processes. There are three major reasons why this control method is desirable:

1) In an environment which does not support real-time scheduling, a table driven RTTC can enforce timely delivery of packets. Thus, the RTTC can work consistently with the CAC to ensure that all the deadlines are satisfied.

2) Table driven mechanism can regulate traffic of a connection so that the network entry traffic can be less bursty at the entrance of the network, while guaranteeing deadline at the same time. Consequently, this reduces the disturbance caused by this connection and increases the probability of admission of future connections.

3) Synchronization can be achieved within a connection or among connections since information regarding resources allocated to all the connections is readily available in the execution table. This is a desirable feature of many embedded mission critical systems.

There are three issues, as discussed next, to be addressed in the TD method in order to provide a complete CAC solution.

4.2 Time Slot Allocation

The number of slots allocated to a connection depends on several factors: its deadline as well as the loads of the hosts and the network. Our allocation scheme takes these factors into account. During CAC, the deadline D of a connection is partitioned into three parts: the sender host sub-deadline D_{sender}, the network sub-deadline $D_{network}$ and the receiver host deadline $D_{receiver}$. The number of time slots allocated to a connection on a host is inversely proportional to its sub-deadline. To ensure a fair deadline partition, we first define and calculate a load parameter that reflects the relative load of the individual parts. The deadline is then partitioned in proportion to the load parameters. The host with a heavier load is given a larger sub-deadline, thus relatively balancing load among the hosts. The detailed partition method is not presented here due to space limitation, but can be found in [10].

4.3 Host Delay and Buffer Requirements

For a connection on a particular host, we introduce two functions: the minimum service function $S_{min}(I)$ and the maximum service function $S_{max}(I)$. They are defined as:

$S_{min}(I)$ = Min. amount of traffic sent for a connection at the host during any interval of length I. (7)

$S_{max}(I)$ = Max. amount of traffic sent for a connection at the host during any interval of length I. (8)

Let $\Gamma(I)$ be the maximum rate function of the traffic submitted by the application to NexEx at the sender host. Using the similar arguments as used for equation (4), the worst case buffer length A_{send} in the application buffer of the sender host is given by

$$A_{send} = \max(I*\Gamma(I) - S_{min}(I)). \qquad (9)$$

The worst case delay suffered in the sender host is

$$d_{send} = \max(S_{min}^{-1}(I) - F^{-1}(I)). \qquad (10)$$

where $S_{min}^{-1}(I)$ and $F^{-1}(I)$ are the inverse functions of $S_{min}(I)$ and $F(I) = I*\Gamma(I)$, respectively. For a complete proof, see [10]. Delay and buffer requirement at the receiver can be derived similarly. We omit them due to space limitation.

4.4 Traffic Description at the Entrance of the Network

The traffic function $\Gamma'(I)$ measured at the system buffer is given by

$$\Gamma'(I) = \min(S_{max}(I), (I + d_{sender}) * \Gamma(I + d_{sender}))/I \quad (11)$$

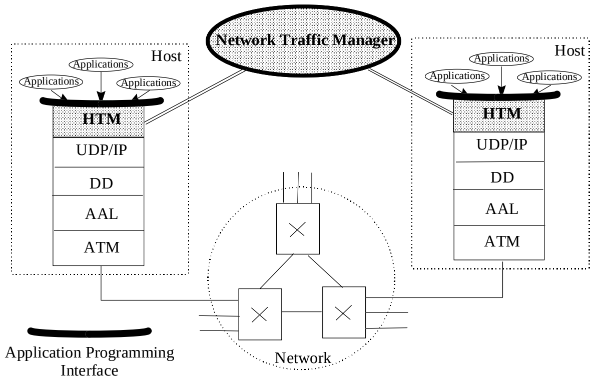

Figure 1 NetEx Architecture

Equation (11) is a direct consequence of delay property of maximum rate function [1] and the fact that $S_{max}(I)$ is a trivial upper bound on the output traffic. We assume that the delay in the protocol stack of the host is constant. So $\Gamma'(I)$ is the same traffic that shows up at the entrance of the AAL layer. The traffic then goes through the AAL and ATM layers before being transmitted into the network. To obtain the description of traffic at the entrance of the network, we should analyze the impact of processing at AAL and ATM layers and then convert $\Gamma'(I)$ accordingly. This process has been discussed in [12].

5. Implementation and Performance

5.1 The Architecture

We have implemented and tested a version of NetEx on an ATM LAN with five ATM switches and Sun Sparc 4 workstations running Solaris 2.5. In the current implementation, NetEx consists of two major components, namely Host Traffic Manager (HTM) and Network Traffic Manager (NTM) (Figure 1). Every participating host has an HTM module resident in it. HTM at a local host is responsible for managing connection and traffic activities on the host. NTM, on the other hand, is responsible for managing the activities network wide. NTM can be located at a central host or it can be implemented in distributed manner. Our current implementation has a centralized NTM.

5.1.1 The Network Traffic Manager

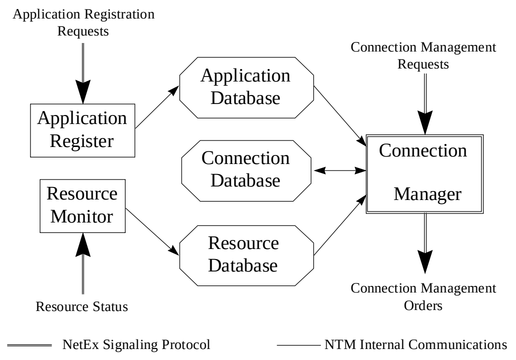

Figure 2 Architecture of the Network Traffic Manager

The Network Traffic Manager (NTM) module in NetEx is primarily responsible for management of connections in the entire system. All the requests for connection establishment go through NTM, where the decision to admit or reject the connection is made and resources to the accepted connections allocated. In addition to connection management, the NTM also maintains a database of applications that are registered in NetEx. Thus, the NTM has global knowledge about the entire system. Figure 2 shows the architecture of the NTM. Basically, the NTM consists of three databases and three execution modules. We briefly describe their main functions:

- *Application Database and Application Register*. Application here refers to a user program that uses communication service in the system. To protect the system from misbehaving applications, NetEx requires all the applications to be registered. The information to be registered includes: application's name, location, and rights (which allow an application be a source, receiver, and/or initiator of a connection). The *application register* is responsible to process the registration and store it in the *application database.*
- *Resource Database and Resource Monitor*. The admissibility of a connection depends on the availability of resources. For example, a route should not be selected if a link in the route is broken. The *resource monitor* utilizes the built-in mechanism in ATM to periodically collect status information of resources and store it in the *resource database* which will be utilized for connection admission.
- *Connection Database and Connection Manager*. The

function provided by the *connection manager* includes the connection admission control and connection tear-down. The *connection database* maintains a list of records for the connections that are currently active. The CAC is the most critical module in NetEx. We have discussed its functionality and its relationship with the run-time traffic control in Section 2.

5.1.2 The Host Traffic Manager

Host Traffic Manager(HTM), as the name implies, performs traffic management at the host. Traffic management at hosts in tandem with system wide connection admission control is thus vital for providing delay guaranteed communication service at the application layer. As we discussed in Section 2, a proper integration of host traffic control with connection admission has been a key design strategy in NetEx. The HTM architecture is shown in Figure 3. It consists of several modules that work cooperatively in order to realize the HTM functionality. These modules are:

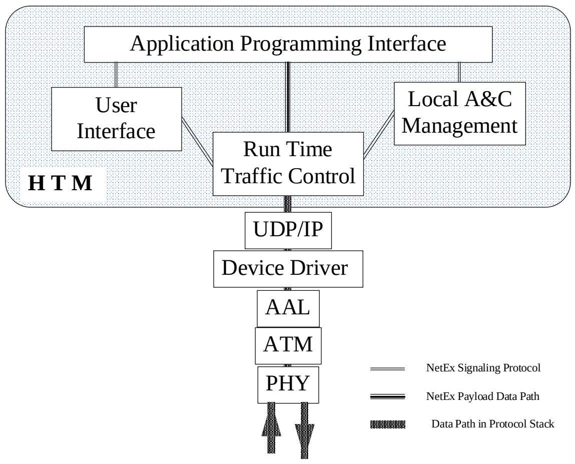

Figure 3 HTM Architecture and its Position in the Protocol Stack

- *Run Time Traffic Control Module (RTTC).* RTTC is the module that is responsible for carrying out the host traffic control as per the instructions of the NTM. The RTTC is the execution engine in the HTM that monitors connection traffic and regulates the packets belonging to different connections in a host.

- *Local Application and Connection Management Module* (LACM). While the decisions on connection admission are made by the NTM, the HTM has to provide necessary local support for the NTM to function efficiently. The needed support includes (i) information forwarding between applications and the NTM (e.g., CAC requests from applications to the NTM and CAC decisions from the NTM to applications), (ii) estimation of host delays (i.e., d_{sender} and $d_{receiver}$), (iii) local parameter setting (e.g., buffer location and size, traffic transmission rates, etc.), and (iv) local bookkeeping.

- *User Interface Module.* A user interface is a desired feature for monitoring and control. The User Interface Module serves as a graphical interface with point-and-click mechanisms to manage connections, monitor host and network status, collect traffic statistics, record connection history, serve as a debug console, etc.

- *Application Programming Interface.* An application programming interface provides a library of routines that an application uses to avail NetEx services. This interface includes routines for requesting connection establishment, connection tear down, and data transfer.

5.2 Performance and Observations

The performance metric of interest is admission probability. For a given session, the admission probability (AP) is defined as the ratio of the total number of connections admitted to the total number of connection establishment requests submitted. We report findings on the relationship between AP and number of connections generated per host.

The data furnished here are based on two popular benchmarks which have been widely used in studies of this nature. The data sets are MPEG-1 encoded VBR video traces. The first benchmark is from the movie *Star Wars* [14] and the second is from a video conferencing trace [15]. The user application presents one MPEG frame every 40ms in each connection. The basic layout of our experiment consisted of two workstations connected by one ATM switch. One of the workstations sends data to the other workstation which sends the same data back immediately to the sender workstation. The round trip delay is measured and the deadline is compared against this delay. Every connection sends the same benchmark traffic (the Star Wars or the video conference). The deadlines of the connections are exponentially distributed with mean 50ms and 100ms. We then test to see how many connections out of all the requested connections can be admitted without violating the corresponding deadlines. The RTTC used the table-driven method with slot sizes chosen to be 10ms and 20ms.

The collected data are plotted in Figure 4. The ordinate is the admission probability and the abscissa is the traffic in demand in terms of the number of connection establishment requests generated. From this figure, we observe that AP improves as the mean increases from 50ms to 100ms. Netex has also some unique behavior :

- The admission probability does not start from 100%. That is, even for a set of one connection, the admission probability is less than 100%. This is due to the fact that there is a nonzero minimum delay caused by a table driven method utilized by the RTTC and other constant servers in the system. A connection has to be denied admission if its deadline is less than this minimum delay. The admission probability stays almost constant when this minimum delay is the only limiting factor.

- For small number of connections, the admission probability corresponding to 10ms slot size is higher than that corresponding to 20ms slot size. This is because for smaller slot size, the delay is also smaller. However, as the number of connections increases, admission probability for 20ms slot size becomes higher than that for 10ms slot size. This can be explained as follows. The RTTC, as a run-time traffic manager and server, consumes CPU time. For a smaller slot size, the RTTC will be awoken more frequently and will consume more CPU time. Hence the admission probability for 10ms slot size drops below that of 20ms slot size. In general, larger slot size tends to give a more stable performance.

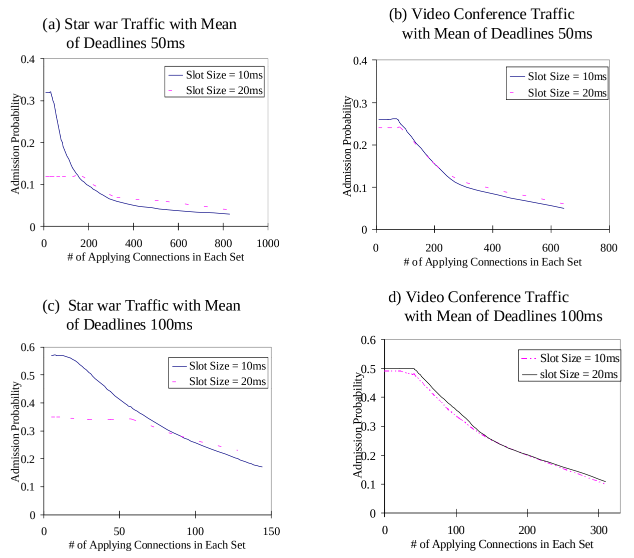

Figure 4 Admission Probability vs. Traffic in Demand

6. Summary

We have described the design and implementation of a software tool kit, NetEx, that aims at providing deadline-guaranteed connection-oriented communication service at the application layer. The key contributions which make our work innovative and unique are as follows:

- By integrating system connection admission control with host run-time traffic control in a consistent manner, we have developed techniques to guarantee message deadlines at application level. While our current implementation involves ATM networks, the techniques we developed are applicable to any switched local area network used by mission critical systems.

- Our tool kit is flexible and easy to use. The run-time traffic control schemes allow NetEx to meet the disparate needs of different applications. The modular design and implementation, and the support provided by the API and the graphical user interface also exemplify this claim.

- NetEx is realized with network products which are currently commercially available and does not require any change. Hence, the technology we have developed is immediately applicable to the design and implementation of mission critical systems which require connection-oriented real-time communication at the application layer.

7. Acknowledgement

This work is partially sponsored by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number F49620-96-1-1076 and by Texas Higher Education Coordinating Board under its Advanced Technology Program, grant number 999903-204.

8. References

[1] R. L. Cruz. A Calculus for Network Delay, Parts I&II. *IEEE Transactions on Information Theory*, January 1991.

[2] D. Ferrari. Real-Time Communication in an Internetwork. *Journal of High Speed Networks*, Jan. 1992.

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

[4] A. K. J. Parekh. *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks*. Ph.D. Thesis, MIT, Cambridge, MA, 1992.

[5] A. Raha, S. Kamat, and W. Zhao. Guaranteeing End-to-End Deadlines in ATM Networks. *Proc. of the 15th IEEE Int. Conf. on Distributed Computing Systems*, June 1995.

[6] A. Mehra, A. Indiresan, and K.G. Shin. Resource Management for Real-Time Communication: Making theory meet Practice. *Proc. of the Real-Time Applications and Technology Symposium*, Brookline, MA, June 1996.

[7] H. Zhang. Providing End-to-End Performance Guarantees Using Non-Work-Conserving Disciplines. *Computer Communications*, 1996.

[8] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new resource ReSerVation Protocol. *IEEE Network Magazine*, September 1993.

[9] ATM Forum. *ATM User-Network Interface Specification Version 3.1*, 1994.

[10] C. Li. *Design and Implementation of Network Traffic Management*. M.S. Thesis, Texas A&M University, College Station, TX, 1997. *(In preparation)*

[11] A. Raha, S. Kamat, and W. Zhao. Admission Control for Hard Real-Time Connections in ATM LANs. *Proceedings of IEEE Infocomm*, March 1996.

[12] F. Feng. *Integrated Traffic Control and Management for Hard Real-Time Applications in High Speed Networks*. Ph.D. Thesis. Texas A&M University, 1996.

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

[14] M. W. Garrett, and W. Willinger. Analysis, Modeling and Generation of Self-Similar VBR Video Traffic. *Proceedings of ACM SIGCOMM*, London, Sept. 1994.

[15] O. Rose. *Statistical Properties of MPEG Video Traffic and their Impact on Traffic Modeling in ATM Systems.* University of Wuerzburg. Institute of Computer Science Research Report Series. Report 101. Feb. 1995.