An Efficient Call Admission Control for Hard Real Time Communication in Differentiated Services Network

Prashant Baronia
Department of Computer Science
Indian Institute of Technology Bombay
Mumbai, India
email: prashant@cse.iitb.ac.in

Anirudha Sahoo
Kanwal Rekhi School of IT
Indian Institute of Technology Bombay
Mumbai, India
email: sahoo@it.iitb.ac.in

Abstract - In this paper we study hard real time (HRT) communication over a differentiated service network. Differentiated Services was proposed by IETF as a scalable QoS architecture [1]. Diffserv service providers can provide delay and bandwidth guaranteed service (as per service level agreement (SLA)) by implementing diffserv services with a fair packet scheduling algorithm like WFQ. But this QoS is provided at an aggregate traffic level. Since HRT application require per flow level delay guarantee, simply deploying HRT application over Diffserv network is not adequate. Additional mechanism is needed to provide per flow level QoS over diffserv network. We have devised a method for calculating end-to-end delay of a connection through a diffserv network based on the theory of Latency Rate (LR) server [2]. Our delay analysis considers actual total incoming traffic to the diffserv network rather than SLA level aggregate traffic. Hence it results in better resource utilization. We propose an efficient call admission control (CAC) based on our delay analysis. Through simulation we show that our method performs better than the traditional SLA level aggregate traffic based approach.

1. Introduction

In a hard real-time (HRT) system, messages have very stringent deadlines. If a packet is delayed beyond its deadline, then the consequence may be catastrophic. Therefore delay is a very crucial Quality of Service (QoS) parameter for HRT system. Hence resources have to be allocated with utmost care to ensure that packets meet their deadlines. When the network infrastructure of a HRT is IP based, system designer has to evaluate various options available for providing QoS. The Integrated Service (IntServ) [3, 4] architecture is an attractive solution for HRT because it provides guaranteed service which can be used to provide deterministic delay bounds for the packets. Unfortunately, IntServ architecture is known to have scalability problem [5]. Hence Differentiated Services (DiffServ) QoS architecture has been proposed by IETF as a scalable solution [1]. It is aimed at supporting service differentiation for aggregated traffic. Differentiated services distinguishes between a small number of forwarding classes and resources are allocated to those forwarding classes. Diffserv service provider usually has a service level agreement (SLA) with the customer, which, among other things, describes the aggregate traffic expected from the customer network and QoS expected by the customer network from the service provider. Consequently, the diffserv network does not provide QoS guarantees at a per-flow level as is the case in Intserv. Rather, the QoS is provided at an aggregate SLA level traffic. Due to scalability, DiffServ is the architecture of choice at the backbone of a network. Thus, when applications are run over a Internet Service Provider's (ISP) backbone, (e.g., two locations of an enterprise which are geographically apart are connected through an ISP), additional mechanism has to be put in place to provide per-flow QoS. When such applications have hard deadlines, then there is an obvious need to provide delay guaranteed service over the diffserv network. A rocket launching system being remotely operated from a far off command control over a diffserv network is a good example of such an application. The rocket launching application would require a delay guaranteed service from the diffserv network. The diffserv service provider can provide delay guaranteed service to aggregate traffic by implementing diffserv service using some fair packet scheduling algorithm like WFQ [6]. However, for the hard real-time application delay guarantee at the aggregate level is not good enough. Hence, a mechanism to provide delay guaranteed service at a per flow level in the diffserv network is needed. A traditional way to achieve this is to admit individual connections based on delay calculation of SLA level aggregate traffic. But this would translate to overallocation of resources for each connection, which would lead to low

resource utilization.

In this paper, we present a more efficient method to admit individual connections into the differv network only when their individual deadlines can be met across the diffserv network. We have devised a method for calculating end-to-end delay of a connection through a diffserv network based on the theory of Latency Rate (LR) server [2]. Our delay calculation is not based on SLA level aggregate traffic, but it is calculated considering the actual total traffic from the customer network at the time of delay calculation. Thus, our method achieves better resource utilization. We assume that the packets from HRT application are marked as one of the forwarding classes for which the diffserv service provider guarantees a certain bandwidth and worst case delay by having the packets of the forwarding class scheduled by a fair queuing scheduler like WFQ which belongs to LR class of server. Since most of the contemporary routers now support WFQ or some variant of it in their scheduling architecture, we think it is a reasonable assumption. We then devise an efficient call admission control (CAC) algorithm which uses our delay analysis to admit calls so that per flow end-to-end deadlines are met. Through simulation we show that our approach performs better than the traditional method based on SLA level aggregate traffic for delay calculation in terms of admission probability thus achieving better resource utilization.

Real-time communication have been studied for CSMA/CD [7], token ring [8] and ATM networks [9]. Admission Control algorithms has been studied for ATM network in [10], for ATM based heterogeneous networks in [11, 12]. A method for providing absolute differentiated service for real time application in network that use static priority scheduling has been studied in [13].

2. System Model

2.1. Service Level Agreement

We assume that there is a service level agreement (SLA) between the customer and the diffserv service provider. The SLA consists of the following four parameters (all these parameters are at an aggregate traffic level). The first two pertains to customer traffic The third one pertains to both customer traffic as well as service provided to the customer in terms of bandwidth. The last one pertains to the service received by the customer's traffic in terms of latency of receiving the guaranteed bandwidth.

σ^{SLA} : This is the amount of burst that the customer can inject into the diffserv network.

MTU^{SLA} : This is the maximum size of packets that can be sent by the customer network to the diffserv network.

ρ^{SLA} : This is the amount of bandwidth that is allocated for the customer traffic. The customer is also supposed to limit the rate of its incoming traffic to this value. This entity will also be denoted as $\rho^{allocated}$ in this document.

θ^{SLA} : This is the total latency after which the customer traffic is guaranteed to get minimum of ρ^{SLA} bandwidth.

The SLA based aggregate traffic A^{SLA} injected by customer network into the diffserv network in an interval $(\tau, t]$ is bounded by

$$A^{SLA}(\tau, t) \leq \sigma^{SLA} + \rho^{SLA}(t - \tau) \tag{1}$$

So the binding with SLA is that as long as the customer traffic conforms to those traffic parameters (σ^{SLA} , ρ^{SLA}, MTU^{SLA}), the service provider should guarantee the service (ρ^{SLA} , θ^{SLA}).

2.2. Differentiated Services Network

A Diffserv network is a set of nodes, connected by links as shown in Figure 1. Further, these nodes are capable of providing different forwarding treatments to packets belonging to different traffic class in accordance with diffserv specification [1]. There are two kinds of nodes in the network. The nodes which are connected to the customer network are the boundary nodes, which typically do policing and also marking[1] of packets. Traffic in the diffserv network is serviced by LR servers (defined later in this paper) at each node. Customer chooses an ingress and egress boundary node for real time communication.

Between an ingress and egress node, the number of nodes (hence the number of LR servers) a customer's traffic goes through is known. We assume that the bandwidth allocated to the customer's traffic at each node is same and is equal to the SLA bandwidth (ρ^{SLA} or $\rho^{allocated}$) agreed upon between the customer and the service provider. For example,

[1] Classifying a packet as one of the diffserv forwarding class is known as *marking*. The class information is put into the packet header as Differentiated Service Code Point (DSCP) value.

in Figure 1 connections from node A to node D would go through A, B, C and D. Thus, it would go through four LR servers.

2.3. Traffic Description of a Connection

Input traffic of a connection is assumed to be token bucket constrained. The token bucket burst parameter of a connection C_i coming from customer with SLA SLA is denoted by σ_i^{flow} and the corresponding rate is denoted by ρ_i^{flow}. Connection C_i also has an associated deadline $Deadline_i^{flow}$. Thus, the input traffic from a flow or connection C_i presented to the diffserv network in an interval $(\tau, t]$ is given by

$$A_i^{flow}(\tau, t) \leq \sigma_i^{flow} + \rho_i^{flow}(t - \tau) \tag{2}$$

3. Delay Analysis

For HRT system end-to-end delay is one of the most important QoS parameter. So in this section we will provide the delay analysis that is used in our CAC algorithm while admitting connections.

3.1. Latency-Rate Servers

Our delay analysis is based on the concept of Latency-Rate (LR) server. The theory of LR servers provides a means to describe the worst case behavior of a broad range of scheduling algorithms in a simple and elegant manner[2]. The theory of the delay calculations is based on the concept of a *busy period* of a session. The busy period of a session is a period of time during which the average arrival rate of the session remains at or above its allocated rate. For a scheduling algorithm to belong to the LR class, it is only required that the average rate of service offered by the scheduler to a busy session, over every interval starting at time θ from the beginning of the busy period, is at least equal to its allocated rate [2]. The parameter θ is called the latency of the scheduler. All the schedulers that provide bandwidth guarantees *viz.* Weighted Fair Queuing, Virtual Clock, SCFQ, Weighted Round Robin etc. exhibit this property and can therefore be labeled as LR servers. The behavior of an LR scheduler is determined by two parameters: the *latency* and the *allocated rate*. We have chosen LR servers for our system because it provides deterministic delay bound which is required for HRT systems. Also, implementation of scheduling algorithms such as WFQ or its variants which belong to LR class of servers are readily available in networking devices today.

3.2. Definitions and Notations

Most of the definitions and notation we provide here are taken from [2]. In [2] they have given the definitions for individual flow (or session) i. In our context, the definitions and formulae should be applied to SLA level aggregate traffic. We denote ρ_i the rate allocated to the flow i. Let $A_i(\tau, t)$ denote the arrivals from session i during the interval $(\tau, t]$ and $W_i(\tau, t)$ the amount of service received by session i during the same interval. In a system based on the fluid model, both $A_i(\tau, t)$ and $W_i(\tau, t)$ are continuous functions of t. However, in packet-by-packet model, we assume that $A_i(\tau, t)$ increases only when last bit of the packet is received by the server. Likewise $W_i(\tau, t)$ increases only when the last bit of the packet in service leaves the server.

Definition 1 *A **system busy period** is a maximal interval of time during which the server is never idle. During a system busy period the server is always transmitting packets.*

Definition 2 *A **session busy period** of flow i is a maximal interval of time $(\tau_1, \tau_2]$ such that at any time $t \in (\tau_1, \tau_2]$ packet of the session i arrive with rate greater than or equal to ρ_i,*

$$A_i(\tau_1, t) \geq \rho_i(t - \tau_1) \tag{3}$$

Let us define the general class of *LR* servers. A server in this class is characterized by two parameters for each session i it services: *latency* θ_i and the *allocated rate* ρ_i. Let us assume that the jth busy period of session i starts at time τ. We denote $W_{i,j}^S(\tau, t)$ as the total service provided by the server S to the traffic of the session that arrived during the jth busy period until time t.

Definition 3 *A server S is said to be LR server if and only if for all time t after time τ that the jth busy period started and until the packets that arrived during this period are serviced,*

$$W_{i,j}^S(\tau, t) \geq max(0, \rho_i(t - \tau - \theta_i^S)), \tag{4}$$

where θ_i^S is the minimum non-negative number that satisfies the above inequality.

3.3. Analysis of a Sequence of *LR* Servers

As a connection goes through multiple network devices in the diffserv network and each network device schedules this connection by a LR server, we need to do delay analysis of a connection going through a sequence of LR servers. The input traffic to the first LR server is token bucket constrained.

It has been proved in [2] that the end-to-end delay of a connection i going through sequence of k LR servers is given by

$$D_i \leq (\sigma_i/\rho_i) + \sum_{j=1}^{k} \theta_i^{S_j} \tag{5}$$

The above equation gives delay bound for a connection whose input traffic is token bucket constrained. In our context, the input traffic to the diffserv network is token bucket constrained, but the traffic is the SLA level aggregate traffic coming from the customer network. So the challenge is to provide delay guarantee for per flow connection. A traditional way to achieve this is to admit an individual connection based on delay calculation at SLA level aggregate traffic. In this case, σ_i, ρ_i, and $\sum_{j=1}^{k} \theta_i^{S_j}$ can be replaced by σ^{SLA}, ρ^{SLA} and θ^{SLA} respectively in equation (5) to get the delay bound of a connection. But that would mean that the delay calculation is done assuming as if customer is presenting traffic equal to the SLA traffic. Hence, when the real aggregate traffic from customer is less than the SLA traffic, that would result in overallocation of resources to the individual connections and consequently low resource utilization. A better approach is to calculate end-to-end delay bound based on actual total per-flow traffic, which would lead to high resource utilization. In the next section we show how such a delay analysis can be done.

3.4. Delay Analysis of a Connection in a DiffServ Network

In this section, we present the delay analysis for an individual connection or flow C_i from customer with Service Level Agreement SLA coming in to the diffserv network with token bucket parameters σ_i^{flow} and ρ_i^{flow}. Here we calculate the maximum delay of the connection if it were to be admitted.

Theorem 4 *Let C_i be the incoming connection to the diffserv network from customer with SLA SLA, with token bucket parameters σ_i^{flow} and ρ_i^{flow}. Let Φ be the set of existing connections from the same SLA, then the maximum delay D^{actual} of the connection going through a path in diffserv network consisting of k LR servers $S_1, S_2, \cdots S_k$ in sequence is given by*

$$D^{actual} \leq (\sigma^{actual}/\rho^{allocated}) + \sum_{j=1}^{k} \theta_i^{S_j} - MTU^{SLA}/\rho^{allocated}, \tag{6}$$

where $\sigma^{actual} = \sum_{j \in \Phi} \sigma_j^{flow} + \sigma_i^{flow}$ and $\rho^{allocated} = \rho^{SLA}$.

Proof: Let D_1^{actual} be the delay experienced by a packet that arrived at time t^* during the jth busy period at LR server S_1 which has a latency of θ^{S_1}. So the packet finished servicing at time $t^* + D_1^{actual}$. Hence, the amount of service offered to the session until time $t^* + D_1^{actual}$ is equal to the amount of traffic that arrived until time t^*. That is, if t is the beginning of the jth busy period then

$$W_{i,j}^{S_1}(t, t^* + D_1^{actual}) = A_i(t, t^*) \tag{7}$$

From the definition of *LR* server (equation (4)),

$$W_{i,j}^{S_1}(t, t^* + D_1^{actual}) \geq \rho^{allocated}(t^* + D_1^{actual} - t - \theta_i^{S_1}) \tag{8}$$

The arrival rate $A_i(t, t^*)$ is bounded by

$$A_i(t, t^*) \leq \sigma^{actual} + \rho^{actual}(t^* - t) \tag{9}$$

Hence, from equation (8) and (9) we have

$$\sigma^{actual} + \rho^{actual}(t^* - t) \geq \rho^{allocated}(t^* + D_1^{actual} - t - \theta_i^{S_1}) \tag{10}$$

that is,

$$D_1^{actual} \leq (\sigma^{actual}/\rho^{allocated}) + \theta_i^{S_1} - (t^* - t)(1 - (\rho^{actual}/\rho^{allocated})) \tag{11}$$

Since $\rho^{actual} \leq \rho^{allocated}$ it follows from equation (11) that

$$D_1^{actual} \leq (\sigma^{actual}/\rho^{allocated}) + \theta_i^{S_1} \tag{12}$$

It has been proved in [14] that if τ is the start of the jth busy period at the first server for session i in a sequence of LR servers, if ρ_i is the minimum bandwidth allocated to session i along all the LR servers, the service offered to the packets of the jth busy period after kth LR server is given by

$$W_{i,j}^{S_k}(\tau, t) \geq max(0, \rho_i(t - \tau - \sum_{j=1}^{k} \theta_i^{S_j})) \tag{13}$$

Using equation (13) we can treat the sequence of LR servers as equivalent to a single LR server with latency equal to the sum of their individual latencies. Hence, it follows from equation (12) that

$$D^{actual} \leq (\sigma^{actual}/\rho^{allocated}) + \sum_{j=1}^{k} \theta_i^{S_j}, \tag{14}$$

In [2] a tighter delay bound is obtained by observing that on the last node, latency can be determined based only on the instant of time just after a packet was serviced from the session. This results in a delay bound which would be less by $MTU^{SLA}/\rho_{allocated}$ from that given in equation (14). Then equation (6) follows from (14). Q.E.D.

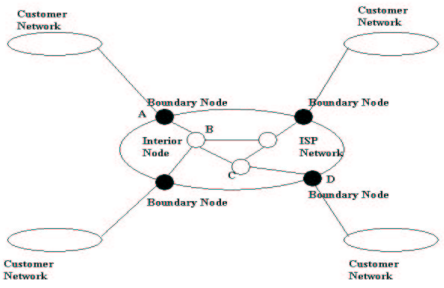

Figure 1: Differentiated Services Network

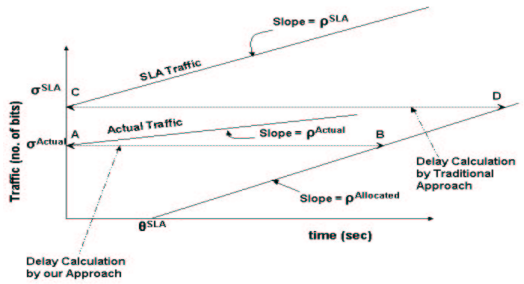

Figure 2: LR server traffic and delay calculation

If we consider a WFQ LR server, then from [2] latency of each one of these servers is $MTU^{SLA}/\rho_{allocated} + MTU^{SLA}/r$, where r is the service rate of each server. Hence if there are k such servers then from (6), the delay bound is given by

$$D^{actual} \leq (\sigma^{actual}/\rho^{allocated}) + (k-1) \cdot MTU^{SLA}/\rho^{allocated} + k \cdot MTU^{SLA}/r \tag{15}$$

Figure 2 shows the delay bound calculation by the two approaches. AB in the figure gives the delay bound using our approach whereas CD is the delay bound using traditional approach. Equation (6) and (15) will be useful for diffserv service provider to specify θ^{SLA}. For example if its diffserv network has WFQ based LR servers then it can advertise θ^{SLA} as equal to $(k-1) \cdot MTU^{SLA}/\rho^{allocated} + k \cdot MTU^{SLA}/r$. If it has some other flavor of LR server implementation then it should calculate θ^{SLA} by substituting for latency of that LR server from the table given in [2]. Thus, the maximum delay of a connections can be generalized to

$$D^{actual} \leq (\sigma^{actual}/\rho^{allocated}) + \theta^{SLA} \tag{16}$$

4. Call Admission Control (CAC)

In this section we devise a call admission control (CAC) algorithm which will be used to admit connections. When a new call comes in, our CAC checks to see that by admitting the new call, all the existing calls as well as the new call can meet their respective delay requirements. The end-to-end maximum delay calculation used in the previous section

is at the heart of our CAC. This CAC algorithm will be executed when a new connection C_i from SLA SLA with token bucket parameters σ_i^{flow}, ρ_i^{flow} comes to the system which has existing set of connection Φ. The service guarantee parameters from diffserv service provider are ρ^{SLA}, θ^{SLA}. Connection C_i has a deadline of $Deadline_i^{flow}$.

Procedure 1 CAC_diffserv(σ_i^{flow}, ρ_i^{flow}, θ^{SLA}, ρ^{SLA}, $Deadline_i^{flow}$, Φ)

1: /* one time initialization and remember the values for the next invocation*/
2: static $\sigma^{actual} = \rho^{actual} = 0$;
3: $\sigma^{old} = \sigma^{actual}$; $\sigma^{new} = \sigma^{old} + \sigma_i^{flow}$; $\rho^{new} = \rho^{old} + \rho_i^{flow}$;
4: **if** $\rho^{new} > \rho^{SLA}$ **then**
5: goto 17
6: **end if**
7: D^{actual} = maximum delay of a packet (using equation (16)) ;
8: **for all** $j \in \Phi$ **do**
9: /* if any of the existing connection misses its deadline reject this connection */
10: **if** $D^{actual} > Deadline_j^{flow}$ **then**
11: goto 17
12: **end if**
13: **end for**
14: **if** $D^{actual} \leq Deadline_i^{flow}$ **then**
15: admit the connection; $\sigma^{actual} = \sigma^{new}$; $\rho^{actual} = \rho^{new}$;
16: **else**
17: reject the connection; $\sigma^{actual} = \sigma^{old}$; $\rho^{actual} = \rho^{old}$;
18: **end if**

This CAC algorithm provides delay guarantees to connections at a per flow level, while it is very simple and efficient. The running time complexity of this algorithm is $O(N)$, where N is the number of connections in the system. Thus, we believe it is very appropriate for had real time communication over diffserv network. Note that *Procedure 1* can also be used admit connection for the traditional method except that equation (5) should be used with appropriate substitution mentioned in Section 3.3 in line number 7 instead of equation (16).

5. Performance Evaluation

In this section we evaluate performance of our CAC algorithm. For this purpose, we have simulated a diffServ network using ns2 simulation software [15]. We have assumed that customer's connection goes through three diffserv enabled network devices each of which provide diffserv service to the customer's traffic using WFQ scheduling. Each link in the diffserv network has a capacity of 1.5 Mbps and has a propagation delay of 1ms. Each connection from the customer network comes with a token bucket constrained traffic whose burst is 160 bytes and rate is 8 kbps and a deadline of 100ms. Connections arrival follows a Poisson distribution. Average number of arrival of connections per unit time is varied to vary the corresponding link utilization. A connection, once admitted, has a lifetime that is exponentially distributed. The average connection lifetime is taken as 180 seconds.

The performance metric used is Admission Probability (AP) which is measured at various link utilization. We define Admission probability as the ratio of total number of connections admitted by the CAC to total number of connections requested.

Figure 3 shows the admission probability versus link utilization when MTU size is 536 bytes. We have chosen 536 bytes for this experiment since it is the minimum MTU size for the Internet [16]. We have measured the admission probability using our approach as well as using the traditional SLA level aggregate traffic method. As is expected, as link utilization increases, admission probability decreases for both the methods. But our method clearly outperforms the traditional method for any link utilization value.

Figure 4 shows the admission probability versus link utilization for various MTU size when our approach is used. As MTU size increases admission probability decreases for a given link utilization. This is so because of the fact that as MTU size increases, packets encounter higher delay at WFQ scheduler. This leads to lesser number of connections being admitted. This result can be used by diffserv service provider to determine what MTU value should be used in customer SLA. For example, if the service provider wants admission probability to be close to 0.85 for utilization up to 30%, then the customer's MTU should be set to 536 bytes or less.

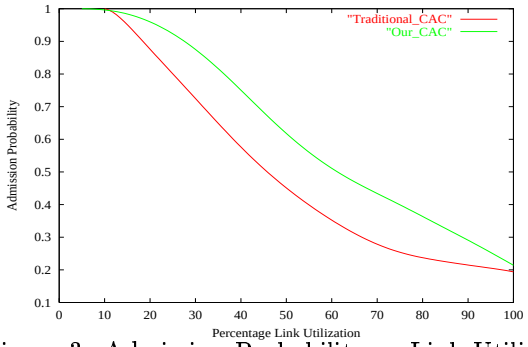

Figure 3: Admission Probability vs Link Utilization for MTU size 536 bytes

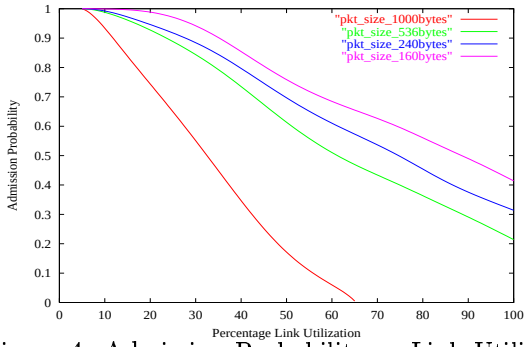

Figure 4: Admission Probability vs Link Utilization using various MTU size with our approach

6. Conclusion and Future Work

HRT systems require per flow level QoS guarantee. Diffserv network only provides QoS at SLA aggregate level. We have provided a methodology by which per flow level delay guarantee can be provided for HRT applications. Traditional way of providing delay guarantee is to calculate delay bound based on SLA aggregate level. We have argued that this method leads to lower resource utilization. A better way is to calculate delay bound based on actual total traffic at the time. We have proposed an efficient CAC based on this method and have shown that our method is better than the traditional method in terms of admission probability. Hence our approach results in better resource utilization.

In this research, we have only considered one class of traffic. It can be extended to include more number of classes, so that when a connection is rejected due to lack of resources of one class of traffic, could possibly be admitted at a lower class if its deadline can be met. This will result in even better performance. Our work can also be extended to include an intserv network at the customer network site. Thus providing end-to-end delay guarantee to a connection going through an intserv-diffserv-intserv can be studied.

References

[1] S. Blake, D. Blake, and M. Carlson, "An architecture for differentiated services," *Internet Request for Comment, RFC 2475*, December 1998.

[2] D. Stiliadis and A. Varma, "Latency-rate servers: A general model for analysis of traffic scheduling algorithms," *IEEE/ACM Transactions On Networking*, vol. 6, October 1998.

[3] R. Braden, D. Clark, and S. Shenker, "Integrated services in the internet architecture: an overview," *Internet Request for Comment, RFC 1633*, June 1994.

[4] R. Braden, "Resource reservation protocol (rsvp)," *Internet Request for Comment, RFC 2205*, September 1997.

[5] Z. Wang, *Internet QoS : Architectures and Mechanisms for Quality of Service.* Morgan Kaufmann Publishers, San Francisco, 2001.

[6] A. Parekh, "A generalized processor sharing approach to flow control in integrated services networks," *MIT Technical Report LIDS-TH-2089*, February 1992.

[7] J. Kurose, M. Schwartz, and Y. Yemini, "Controlling window protocols for time-constrained communication in a multiple access environment," *Proceedings of the 8th IEEE International Data Communication Symposium*, October 1983.

[8] G. Agrawal, B. Chen, W. Zhao, and S. Davari, "Guaranteeing synchronous message deadlines in high speed token ring networks with timed token protocol," *IEEE ICDCS*, January 1996.

[9] A. Mehra, A.and Indiresan and K. Shin, "Resource management for realtime communications: Making theory meet practice," *Proceedings of RTAS'96*, June 1992.

[10] A. Raha, S. Kamat, and W. Zhao, "Admission control for hard real-time connections in atm lans," *IEEE INFOCOM*, March 1996.

[11] B. Chen, A. Sahoo, W. Zhao, and A. Raha, "Connection-oriented communications for real-time applications in fddi-atm-fddi heterogeneous networks," *IEEE ICDCS*, May 1997.

[12] A. Sahoo, W. Zhao, and W. Jia, "Partition-based admission control in heterogeneous network for hard real-time connections," *Proc. of International Conference on Parallel and Distributed Computing*, October 1997.

[13] S. Wang, D. Xuan, R. Bettatti, and W. Zhao, "Providing absolute differentiated services with statistical guarantees in static-priority scheduling," *IEEE INFOCOM*, 2001.

[14] D. Stiliadis and A. Varma, "Latency-rate servers: A general model for analysis of traffic scheduling algorithms," *Technical Report UCSC-CRL-95-38, U.C. Santa Cruz*, July 1995.

[15] http://www.isi.edu/nsnam/ns/, "The network simulator: ns2,"

[16] J. Mogul and S. Deering, "Path mtu discovery," *Internet Request for Comment, RFC 1191*, November 1990.