Vol. 40 No. 9 Transactions of Information Processing Society of Japan Sep. 1999

Regular Paper

Design and Implementation of a QoS Control Mechanism Underlying Perceptual-Time Channels*

YaoXue Zhang,† Zixue Cheng,†† Xiaochun Wang,† Hua Chen† and Shoichi Noguchi††

This paper presents a Quality of Services (QoS) control mechanism for distributed multimedia computing, and describes the design and implementation of this mechanism, which is based on the RSVP protocol. The purpose of this QoS control mechanism is to provide end users with customized computations according to the QoS demands for effective utilization of network resources. The mechanism establishes, maintains, and deletes end-to-end sessions for different multimedia streams and provides both guaranteed service and controlled load service to end users in perceptual-time, but not real time. We discuss the key components of the QoS control mechanism, such as the architecture of the mechanism, the mathematical definitions of QoS parameters, the implementation formats of the control packets based on the defined QoS parameters and RSVP, and the packet scheduling policy in routers. This paper describes the implementation of a prototyping system for QoS control based on the proposed control mechanism.

1. Introduction

In today's computer networks, with the rapid spread of distributed multimedia applications such as video conferencing, live presentation, video on demand, how to guarantee the QoS (Quality of Services) demanded by end users is becoming one of the most important issues. We consider QoS as a set of parameters such as delay time, jitter, loss ratio, and synchronization that are used to measure the satisfaction of end users of media streams and the utilization of computer networks. A distributed multimedia application is a continuous media stream that passes through one or more networks. It has two important properties[1]: (1) the fidelity is often dependent on the timeliness with which it is presented, and (2) It is often tolerant of the loss of some information content, particularly when it is known how the data is to be used.

These properties impose the requirement that the code for manipulating the media data should be scheduled with suitable windows of time, and some information may not need to be delivered. QoS is used to describe these two properties of media streams.

According to the above properties, if we want to obtain a high-quality media stream with effective utilization of network resources, a QoS control mechanism is needed in network systems[9]. The mechanism for controlling QoS has to handle two major concerns: (1) The QoS demands from network customers for their distributed multimedia applications, and (2) The customized computations on the requested QoS, such as admission control, resource location, and scheduling.

The QoS requirements for media streams are mostly set on a per-user and per-application basis. Even with the same type of application, such as video conferencing, different media streams may require different level of QoS. Consequently, a clear definition for QoS has to be given to measure the differences between media streams and to capture, display, store, and deliver media streams[8]. Some definitions for QoS have been given and specified by the International Standards Organization (ISO)[23] and Internet Engineering Task Force (IETF)[22]. The IETF also defines the services provided to end users of media streams by networks with QoS control, as guaranteed service[11] and controlled load service[10]. Moreover, a resource reservation setup protocol named RSVP[9] has been defined to support resource requirements. However, the situation is that there is still no QoS definition that supports a real implementation

† Department of Computer Science & Technology, Tsinghua University, Beijing, China
†† School of Computer Science and Engineering, University of Aizu

* This work is sponsored by National 973 Fundamental Research Program of China, Chinese National Science Funds, and a Research Grant from Fukushima Prefecture, Japan.

and no architecture that can provide standardized guaranteed service and controlled load service to users in an end-to-end manner. Therefore, wide area networks such as the Internet and local area networks such as Ethernets are still providing best-effort service that can not support the delivery of media streams.

In this paper, we propose an QoS control mechanism, which creates a Perceptual-Time Channel connecting end user with end user, through which media streams can be delivered with the required QoS. This control mechanism is based on the RSVP protocol and the QoS parameters definition proposed in this paper. It provides guaranteed service and controlled load service for end users on demand with the specified QoS parameters. We designed the implementation formats of these QoS parameters and corresponding control packets on the basis of RSVP protocol, so that the QoS requirements of end users and the corresponding control information can traverse a network and be intercepted by routers and hosts in the network.

We also design the structure of the Perceptual-Time Channel (PTC). We consider a PTC as a session layered on the IP protocol with QoS control functions and with the exact resources necessary for required services. To manage the resources, we also provide a packet-scheduling policy and a packet-dropping scheme to allocate the CPU time and buffers of routers. In addition, we describe the design and implementation of an admission control method and an QoS negotiation module for managing the acceptance or rejection of multimedia applications as part of a central control method for establishing, maintaining and removing PTCs.

In comparison with other proposals for implementation of guaranteed service and controlled load service, such as Weighted Fair Queuing (WFQ)[20], we define more QoS parameters and their implementation formats. Our scheduling policy uses these parameters to allocate resources. More specifically, WFQ scheduling is only based on the average delay time and the number of packets lost in a given time. In addition to the above two parameters, our scheduling policy uses the longest length of the bursty packets of a stream, the largest allowable jitter on the delay time, and the probability of packets whose delay times are longer than average delay time, in order to assign a CPU service frame to streams. Therefore, our system can allocate resources more efficiently and accurately.

Section 2 describes the architecture of a PTC, including the functional modules and the logical structure. The QoS parameters for different media streams based on our classification method and the IETF service models are defined in Section 3. The formats of QoS control packets and the control procedure based on RSVP are described in Section 4. The policy and the algorithm for packet scheduling in routers are described in Section 5. Section 6 discusses the QoS negotiation and admission control. Section 7 offers some the concluding remarks.

2. Architecture of a Perceptual-Time Channel

2.1 Structural Overview

The concept of a Perceptual-Time Channel (PTC) is motivated by both the different needs of different media streams and effective utilization of network resources.

A PTC is a session between end users with two different session ports. It is created to satisfy the minimum requirements of a media stream when the network is running short of resources, or to satisfy the user's requirements in full or almost in full when the network has ample resources. Here, we assume that there exists an adjustable range in the QoS requirement of every media stream, so that the resources allocation can be adjusted according to the utilization of network resources. The protocol for establishing, maintaining, and deleting a PTC is the RSVP protocol, which can be found in Braden, et al.[9] or Wroclawski[10]. The control part of the PTC for monitoring the usage of resources, scheduling of resources, classifying streams and admission controls, however, are based on a set of algorithms or policies that we will discuss in the following sections. Therefore, we focus our discussion in this paper on the control part rather than the protocol.

A system overview of a PTC is shown in **Fig. 1**, where the end users transfer a media stream through a PTC consisting of hosts with QoS controllers, IP devices, and IP routers with QoS controllers. Consequently, the PTC is a control-plane component primarily responsible for the creation, modification, and removal of resource reservations associated with different media streams. Note that it is possible to handle several media streams in one PTC if an application has to deliver several media streams at the same time. For example, a video con-

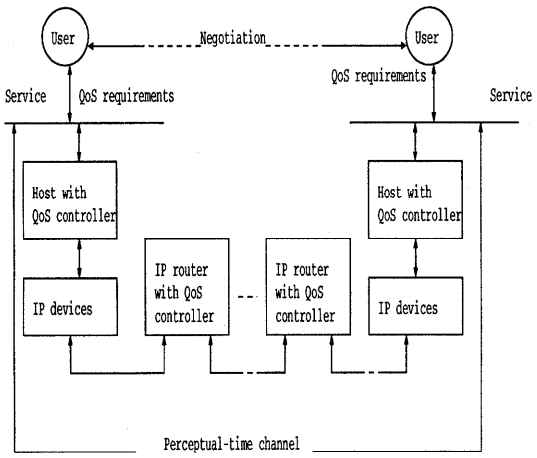


Fig. 1 System overview of a PTC.

ference session may transfer video, audio, and data streams at the same time. Moreover, the PTC can transfer both media stream packets and control packets in the same session along traverse different paths, as in the case described by Barzialai, et al.[17].

The key objectives of building a PTC are (1) to increase the efficiency of usage of resources, (2) to implement the Internet standards, so that existing Internet applications will be compatible with the PTC, and (3) to enable Internet applications to negotiate QoS requirements in an end-to-end manner.

The aim of using resources efficiently is to increase the data transfer performance of the network. Generally speaking, the insertion of a QoS control module into hosts and routers may affect the data transfer performance, since the control and negotiation will produce some overhead in time and space. However, through the control and negotiation, we can reduce the congestion and resource wastage of the network. Moreover, control packets can be transferred through a special control path in the PTC, and thus the overheads can be reduced to a very low level.

However, it is very important to maintain compatibility not only with the current and upcoming standards for QoS negotiation on the Internet, but also with other media network interfaces such as ATM, for interoperability. Our PTC is based on RSVP, and provides guaranteed service and controlled load service, which are IETF standards, as previously mentioned. Moreover, the QoS parameters we will define in Section 3 are compatible with ATM, so it is easy to support applications that use ATM.

It is essential that PTC session creation, resource reservation, resource allocation, admission control, and other processes in hosts and routers should enable Internet applications to negotiate QoS demands in an end-to-end manner. Consequently, the QoS controllers both in end hosts and in routers should be responsible for building PTC session paths with separate control and media flows, in order to reserve and schedule resources, handle error messages, and manage and monitor the usage of resources. In the following subsections, we elaborate the function modules on QoS controllers of both routers and hosts.

2.2 Functional Model of a PTC

In order to guarantee end-to-end QoS for a media stream, a PTC has to include the following functional modules in its host nodes or router nodes: a QoS negotiation module, a QoS renegotiation module, a resource manager, and a resource scheduler. Some of them may be embedded in the kernels of operating systems, and some of them may run as concurrent processes on the given operating systems.

The QoS negotiation module is for negotiation among multiple users and PTCs. It also controls the execution of other modules, and provides adaptive control in which the controller adjusts the user's QoS demands in the light of the observed performance of the PTCs. The classifier catalogs the types of media streams and also forwards these types of applications to the negotiation module to negotiate resources. The admission controller implements the decision algorithm that a router or host uses to make a decision on whether to accept or reject a request.

The resource manager includes two submodules: a computing resource manager and a communication resource manager. The computing resources we consider here include CPUs, memories, I/O buffers, threads, virtual addresses, and stacks, which are all related to the host computer and router, while the communication resources are the network board, output/input ports on the board, communication primitives, protocol components, and multi-cast or unicast addresses. The resource manager is responsible for balancing the user's QoS requirements and the effective utilization of resources. Therefore, policies for calculating the utilization and the satisfaction of users' QoS requirements are necessary. The micro-economic method has been used to resolve this problem[24]. The resource

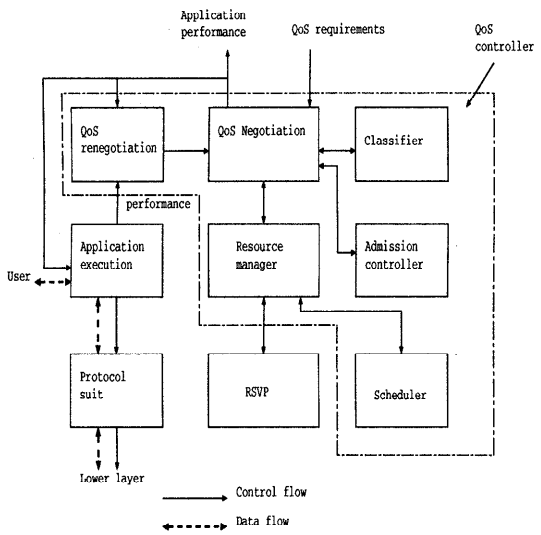


Fig. 2 Functional model of a host computer with a QoS controller.

manager also invokes other modules such as the scheduler, and communicates with the kernel of the operating system.

The resource scheduler is for scheduling and allocating different resources to satisfy the user's QoS demands. Generally speaking, different scheduling policies and algorithms are needed for allocation of different resources. For example, the scheduling method for allocations of buffers and CPU times in routers is different from that in end hosts. Our PTC provides an architecture to include these scheduling algorithms. To show how the PTC provides an end-to-end QoS negotiation for users with guaranteed service or controlled load service, we will give a scheduling algorithm for packet scheduling in Section 5.

Figure 2 shows the functional modules of a host. After the QoS negotiation module receives a QoS request from a user's media stream, it invokes the classifier to classify the media stream, according to the definition of QoS parameters given in Section 3. The QoS negotiation module then invokes the resource manager in order to obtain sufficient resources for the required QoS in the host. The negotiation module also negotiates with routers in the PTC in order to obtain sufficient network resources. The admission controller makes a decision to accept the user's request, if the resources of the host and network are sufficient to guarantee the required QoS. Otherwise, the user's request is rejected, and the admission

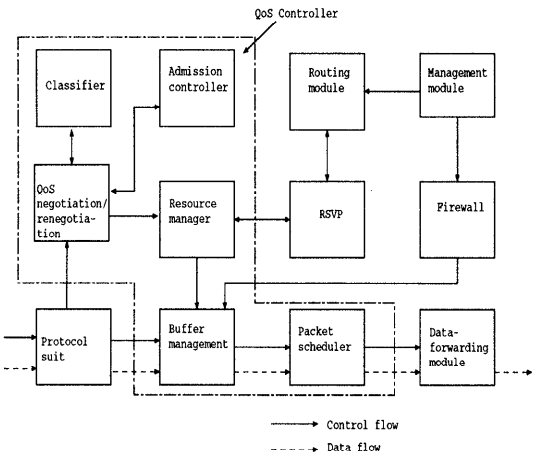


Fig. 3 Functional model of a router with a controller.

controller suggests another QoS level that can be guaranteed to the user. The renegotiation module is called if resources in the host become insufficient or a renegotiation request is received from a router of the PTC on account of a new stream with higher priority trying to share the resources. The renegotiation module negotiates with the user on QoS levels. The scheduler is invoked by the resource manager and used to allocate resources to satisfy the user's QoS requirements.

Figure 3 shows the functional modules of a router in the PTC. When a request for resources to guarantee the QoS of a user's stream is received by the protocol suite, the negotiation module invokes the classifier and the resource manager in order to classify the media stream and to obtain sufficient resources for the required QoS in the router. The negotiation module also negotiates with other routers and end hosts in the PTC, in order to obtain sufficient resources for the PTC. The scheduler allocates resources to process the incoming datagrams. In particular, a packet scheduler in a router allocates the CPU and buffers to the packet queues with higher priorities, so that the forwarding of different media streams can be managed to meet the QoS demands. It also drops packets if the resources become insufficient. Each incoming packet is mapped to a particular class according to the definition of QoS parameters given in Section 3, so that the scheduler can select the streams and schedule resources for incoming packets. The admission controller determines whether a new stream can be granted such that the requested QoS can be guaranteed without affecting existing streams.

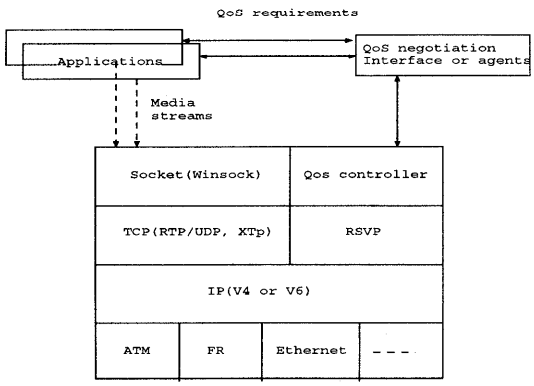


Fig. 4 Layer protocol stack in our PTC.

If a router receives a request for resources from a new stream with a higher priority than an existing stream and the resources are insufficient to guarantee the QoS of both streams, the renegotiation module will be invoked. The module will negotiate with the routers and hosts of the existing stream, and finally with the users of the stream, to see if they can decrease the QoS level. Modules such as the routing module, management module, and firewall are for the general purpose of forwarding data in routers.

In either Fig. 2 or Fig. 3, we divide the data flow from the control flow. This means that although an application has to be executed after ending QoS negotiations, the data flow of one media stream and the control flow for another media stream can be processed in a concurrent processing manner. It also increases the efficiency of the data handling for QoS control[17].

2.3 Protocol Stack Architecture of a PTC

The functional model of a PTC given above is based on a set of protocol components, such as IP (both versions IPv4 and IPv6 can be used), RSVP, and TCP (or RTP, XTP), for delivering media streams. We use this architecture to meet the application requirements of the current Internet. The layered protocol architecture of the PTC is shown in **Fig. 4**. As shown in Fig. 4, communication networks such as Ethernets, frame relay networks, or ATM can be used as transmission media, and these physical networks are in the lowest layer of the PTC protocol socket. The QoS controller is layered over RSVP, and arranged with a socket layer. This is because the QoS controller is responsible for creating, managing, and removing the PTC sessions, and for managing resources in routers and hosts. A control procedure for data flow and control flow can be found in Chen, et al.[5].

3. Definitions of QoS Parameters

To implement the above PTC, we have to define the parameters of QoS mathematically. With these parameters, end users (or agents) can negotiate the required QoS according to the status of network resources. The classification of a media stream and admission control are also based on these parameters. References[10),11)] describe controlled-load service and guaranteed service, respectively. To provide these two kinds of services in our PTC approach, we define the parameters of QoS according to the characteristics and types of media streams, so that the QoS of media streams can be dealt with naturally and distinctly.

[Definition 1] QoS is a tuple $\langle S, F, T, C \rangle$.

Here, $S = a_i, a_i \in \{a_0, a_1, a_2, a_3, a_4\}$ is a symbol for various types of media streams such as hi-fi video/audio, realtime video/audio, non-realtime video/audio, data, and rate-adaptive media streams.

a_0 represents media streams such as hi-fi video/audio streams. It corresponds to media streams with CBR (constant bit rate), which allow very few delays, very little jitter, and a very low loss ratio. Generally, not only a fixed bandwidth and long CPU slots are requested, but also some other resources to guarantee the QoS requirements.

a_1 represents media streams such as realtime video/audio. It corresponds to media streams with R-VBR (realtime variable bit rate), which also require very few delays, very little jitter, and a low loss ratio. However, the speed of a media stream of this type can be changed or varied with time, and such a stream always occurs with bursty transmission.

a_2 represents media streams such as non-realtime video/audio streams. It corresponds to media streams with NR-VBR (non-realtime variable bit rate). In this case, the streams do not require a strong real-time property, but also they occur with bursty transmission.

a_3 represents data streams. In this case, the streams require that the transmission be performed without any error or loss. However, delay or jitter is allowable.

a_4 represents media streams with wide variations in their transmission speeds.

a_0 and a_1 correspond to guaranteed service, while a_2, a_3, and a_4 corrospond to controlled-load service.

$F = \langle d, j, l, p, x_0, x_1, i, b \rangle$ is a vector that describes the performance and flow characteristics of the media streams. Parameters d, j, and l describe the performance, and parameters x_0, x_1, i, and b describe the flow characteristics.

d (integer) represents the largest allowable delay time;

j (natural number) represents the largest allowable jitter on the delay time;

l (actual number, $0 < l < 1$) represents the lost rate, i.e., the number of packets lost in a given time unit;

p (actual number, $0 < p < 1$) represents the probability of packets whose delay times are larger than d;

x_0 (natural number) represents the highest speed ratio of the media stream;

x_1 (natural number) represents the average speed ratio of the media stream;

i (natural number) represents the average cycle time of packets of the media stream;

b (natural number) represents the longest length of the bursty packets.

$T = \langle t_{i1}, t_{i2}, \ldots, t_{ij}, \ldots \rangle$ is a vector that describes the synchronization relations between the current media stream i and auother stream j. Here, t_{ij} $(0 < t_{ij} < 1)$ is an actual number, If stream i does not synchronize with stream j, then $t_{ij} = 0$; otherwise, it will be a number less than 1 or equal to 1 according to the synchronous level of the two streams.

C (natural number) represents the price that the user is willing to pay for receiving the required QoS.

4. Implementation Format of the Control Packets of a PTC

Based on the QoS definition given in Section 3, we give the implementation formats of the control packets for building a PTC. The implementation is developed on the basis of the packet format of RSVP, in order to provide the standardized guaranteed service and controlled load service. A detailed description of the packet format of RSVP can be found in Braden, et al.[9].

We implement 19 classes of RSVP-based control packets as PTC control packets: NULL object (the content of a packet is called an object in RSVP), SESSION object, RSVP_HOP object, TIME_VALUES object, STYLE object, FLOWSPEC object, FILTER_SPEC object, SENDER_TSPEC object and so on. Here, a NULL object is an empty object, a SESSION object defines the communication scope, an RSVP_HOP object is for routing, while a TIME_VALUES object updates the time intervals, and a STYLE object is for specifying forwarding styles. The FLOWSPEC, and FILTER_SPEC, SENDER_TSPEC objects are for specifying the QoS parameters. Specifically, FLOWSPEC describes the QoS requirement, while FILTER_SPEC maps packets into different classes and SENDER_TSPEC describes the flow characteristics of the source. Because of space limitations, we give only the implementation formats of those objects related to QoS parameters.

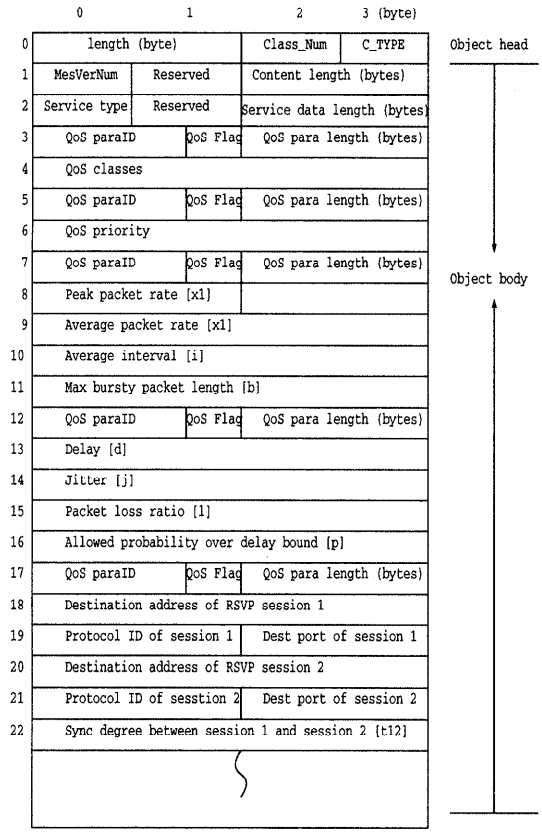


Fig. 5 Implementation format of a FLOWSPEC object.

[Definition 2] Implementation format of a FLOWSPEC object

A FLOWSPEC object consists of a common object head and an object content body whose length does not exceed 65528 bytes, with the format shown in **Fig. 5**.

In Fig. 5, the object FLOWSPEC is an array of words with 32-bit length. The MesVerNum in word 1 identifies the format version of the

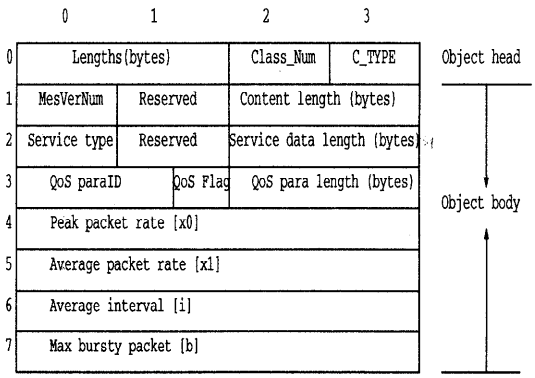


Fig. 6 Implementation format of a SENDER_TSPEC object.

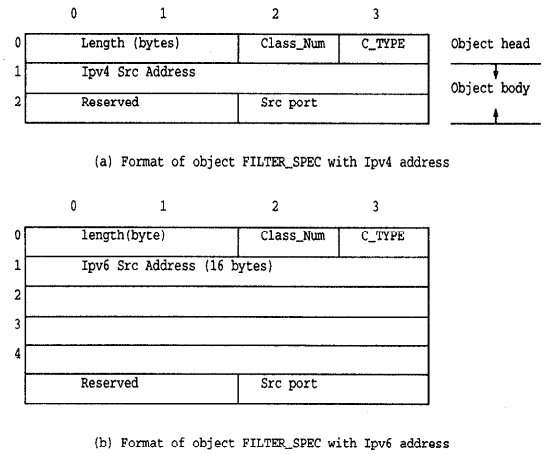


Fig. 7 Format of a FILTER_SPEC object.

packet, the content length is the length of the objects contents, and the service type in word 2 describes the services, such as guaranteed service, and controlled load service. The service data length specifies the words on the specified service, while QoSparaID defines an index for a set of QoS parameters and the *QoS para length* describes the corresponding numbers of bytes of the parameters. Moreover, the QoS classes and QoS priorities classify the type of media stream and assign the corresponding priority. Here, the priority can be updated by a router according to the given scheduling policies. Note that the QoS parameters in the format all follow Definition 1, given in Section 3.

[**Definition 3**] Implementation format of object SENDER_TSPEC

The object SENDER_TSPEC consists of a common object head and an object content body whose length does not exceed 65528 bytes, with the format shown in **Fig. 6**.

The items in Fig. 6 have the same meaning as those in Fig. 5.

[**Definition 4**] Implementation format of a FILTER_SPEC object

A FILTER_SPEC object has two formats for Ipv4 and Ipv6, respectively. Both formats consist of a common object head and different object bodies, as shown in **Fig. 7**.

In Fig. 7, SrcAddress identifies the source IP address of the sender and the Src port identifies the source UDP/TCP port.

In addition to defining the format, we also have to define the corresponding values on the items of the common object head and the paraIDs in order to build our PTC. For example, the class_Num of FLOWSPEC is defined as 9, the SENDER_TSPEC is defined as 12, and the FILTER_SPEC is defined as 10. However, the C_Type of the FLOWSPEC or other objects can take one of the three values 1, 2, and 3 corresponding to its service type such as integrated service, guaranteed service, or controlled load service.

5. Scheduling Policy and Simulation Results

5.1 Packet Queue Scheduling Policy and Dropping Scheme for a PTC

Using the packets defined in Sections 3 and 4 and the RSVP protocol, the end hosts and routers can schedule the necessary resources to build a PTC for media streams or reject an application if the resources are busy. Here, we focus our discussion on routers and present a scheduling policy and dropping scheme for a PTC to allocate the CPU times and buffers.

The basic function of packet scheduling is to reorder the output queues of the packets in a router. Therefore, the scheduling policy has two parts: giving an order to incoming packets for processing and selecting the packets to drop when the buffers are all full. Many scheduling methods have been proposed for managing the output queues and the resulting behavior, such as the priority scheme[21] and Weighted Fair Queuing (WFQ)[20]. However, the priority scheme has the problem that lower-priority classes may be completely prevented from being sent if there are a sufficient number of higher-priority packets. Moreover, though WFQ has been widely used to assign different weights to different incoming queues, deciding of weights for these queues is still difficult.

We give a packet scheduling policy that al-

locates CPU times and buffers by using the QoS parameters given in Definition 1 to schedule the packet queues in routers. These QoS parameters are transferred to the router by FLOWSPEC and SENDER_TSPEC, so that reservation soft states including these parameters can be established in routers.

Our scheduling policy, shown below, is based on QoS requirements, including classes, and delay times.

[Scheduling policy]

Step 1: Decide the weight of each packet queue with a different class.

If a reservation soft state has been established in the router,

Then let r be the service rate of the router CPU and calculate the weights for the different classes of packet following the format:

Weight of packet queue of class a_0 *or* $a_1 = x_0/r + c$

Weight of packet queue of class a_2, a_3 *or* $a_4 = x_1/r$.

Here, x_0 and x_1 are defined as the peak packet rate and average packet rate in Definition 1, respectively. Since classes a_0 and a_1 always request few delays, we give them heavy weights so that they can be processed with the highest priority. Moreover, since the x_1 of some streams with class a_2 may be bigger than the x_0 of streams with class a_0 or class a_1, we have to carefully select the constant c to let the weights of class a_0 or class a_1 streams be heavier.

Step 2: Decide the priority *Pri* of each packet queue, according to the weights and the waiting time of the packet queue.

The priority *Pri* of a packet queue is calculated as follows:

$$Pri = Weight + \alpha_1 \ Waiting \ Time$$

Here α_1 is a constant larger than zero. This formula means that a queue with heavier weight and a longer waiting time will have a higher priority.

Step 3: Assign the CPU service frame (time slots) to the packet queue with highest priority.

The CPU service frame f is calculated as follows:

$$f = \alpha_2 b - \alpha_3 d + \alpha_4 j - \alpha_5 l - \alpha_6 p,$$

where α_2, α_3, α_4, α_5, and α_6 are constants larger than zero, and can be decided according to simulation results, and b, d, j, l, and p are the QoS parameters defined in Definition 1. This formula means that a queue with a larger maximum bursty packet length b, larger jitter j, and smaller delay d, loss rate l, and delay bound p will get a longer CPU service frame f. It also means that the guaranteed service required by a user will get a longer CPU service time.

Step 4: Goto Step 1.

The above scheduling policy is coordinated with a packet-dropping scheme that is used when the buffers of the router become full. We drop those packets with a larger loss ratio, longer delay, and lower priority in turn. The packet-dropping scheme coordinated with the above scheduling policy is as follows:

[Packet-dropping scheme]

If an incoming packet belongs to the class with the largest packet loss ratio l, and the ratio of lost packets is smaller than l,

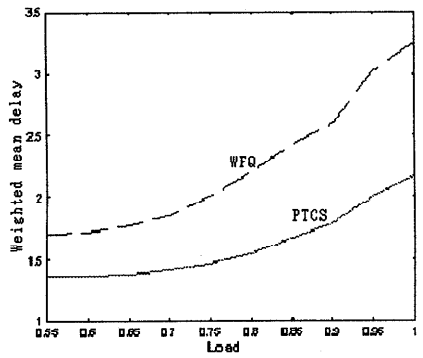
Then drop the incoming packet,

Else look through each packet queue repeatedly until a packet with a larger packet loss ratio l is found, and drop the packet that arrived last.

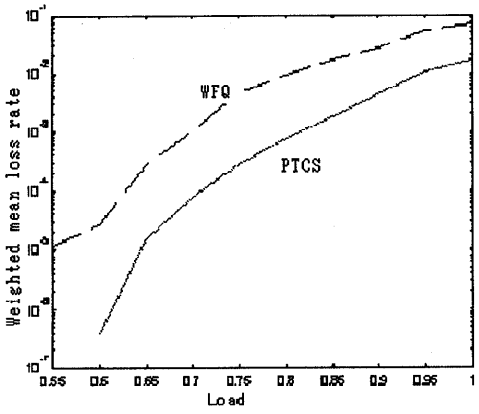
This dropping scheme drops the last-arriving packet with a larger loss ratio.

5.2 Simulation Results and Comparison with WFQ

To evaluate the above scheduling policy and dropping scheme for PTC, we compare it with WFQ in four areas. For simplicity, the scheduling policy and dropping scheme for PTC is called PTCS from now on. First, to qualitatively understand the behavior of these two different algorithms, the weighted mean delay and weighted mean loss rate are determined as functions of the load. Second, we show the performance of one traffic class as a function of the load, under the constraint that the performance requirements of other traffic classes are met. Third, in order to determine the resources required by these two algorithms for achieving certain objectives, the weighted mean loss rate is determined as a function of the buffer size. Finally, we give the throughputs of these two algorithms under various loads. In our simulations, we consider three traffic classes. Class 1 is considered to be constant-bit-rate (CBR) traffic, which requires no delays and no lost packets. Class 2 is considered to be variable-bit-rate (VBR) traffic, which requires few delays and a very low loss rate. Class 3 is considered to be available-bit-rate (ABR) traffic, whose arrival process folows a Poisson distribution.



(a) Weighted mean delay

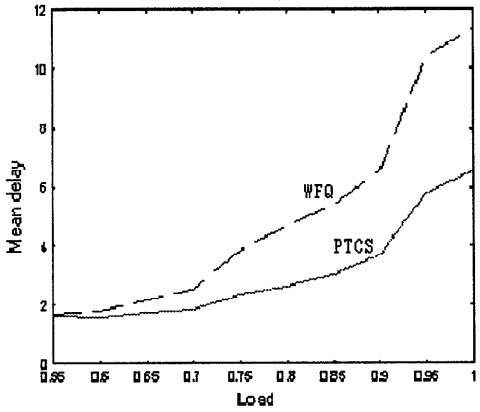


(b) Weighted mean loss rate

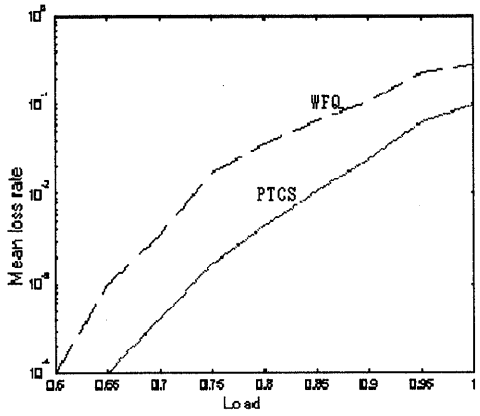
Fig. 8 Performance for three types of traffic.

Weighted mean delay and weighted mean loss rate

As described previously, different types traffic of perceive and therefore measure performance differently. For example, voice (CBR) traffic requires very few delays and a very low loss rate, whereas e-mail (UBR) demands the transmission to be performed without any error or loss, but allows delays. Therefore, we should use weighted mean delay and weighted loss rate to evaluate these two algorithms. The results are shown in **Fig. 8**. It is very obvious that PTCS performs better than WFQ. In Fig. 8 (a), we see that PTCS's weighted mean delay is less than WFQ's, and the difference seems to be greater when the load is heavier. That is to say, in a heavy-load situation, PTCS can provide better performance. Similarly, in Fig. 8 (b), we see that PTCS's weighted loss rate is much smaller than WFQ's.



(a) Mean delay



(b) Mean loss rate

Fig. 9 Performance for certain type of traffic.

Mean delay and mean loss rate of a certain type of traffic

In this simulation, we guarantee to meet the requirements of CBR traffic and VBR traffic, no matter how heavy their loads are. We than measure the mean delay and mean loss rate of ABR traffic. The results are shown in **Fig. 9**, and lead to the same conclusions as Fig. 8.

Resource requirements

To compare the resource requirements of PTCS and WFQ, we fix the load for all three types of trffic and test the weighted mean loss rate for different buffer sizes. The results are shown in **Fig. 10**. From this figure, we can see that PTCS's weighted mean loss rate is smaller than WFQ's for the same buffer size. That is to say, PTCS needs fewer buffer resources than WFQ to achieve the same loss rate.

Throughput

In the last simulation, we compare the throughputs of PTCS and WFQ. The results

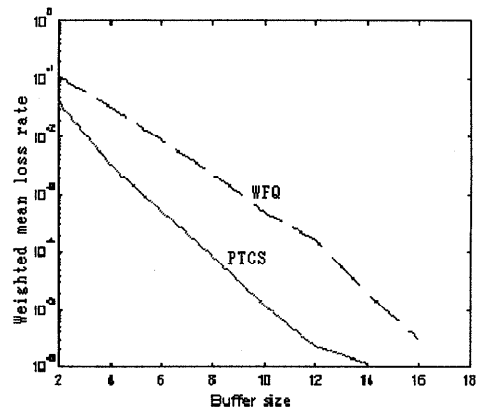


Fig. 10 Resource requirements.

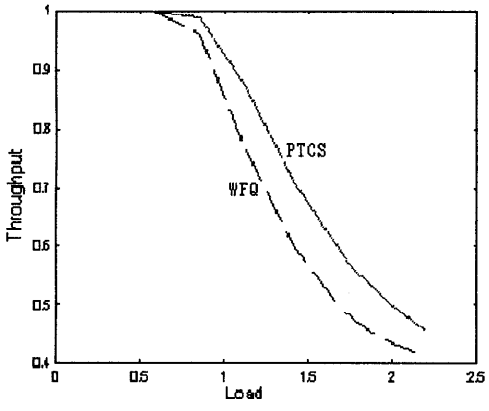


Fig. 11 Throughput.

are shown in **Fig. 11**. It is clear that PTCS's throughput is greater than WFQ's.

6. QoS Negotiation and Admission Control

The negotiation module is closely related to the admission controller. This is because the admission controller module is for deciding whether to accept or reject media streams, while the negotiation module is for negotiating over resource requirements. If the requirements exceed the available resources and the user will not accept a lower QoS level, then the application will be rejected. Otherwise, it will be continually negotiated until it can be accepted by the admission controler. Consequently, we consider both the negotiation module and the admission controller together in this section.

QoS negotiation and admission control require that the routers and the end hosts understand the demands that are currently being made on their assets. Our QoS control approach based on PTC performs QoS negotiation and admission control by a method that programs the end hosts and the routers to measure the actual usage of resources by existing media streams.

There are two methods for managing QoS negotiation and admission control: the segment central control method and the distributed control method. The former uses a server to manage resources and perform admission control at the same time. It also uses agents in each managed host or router to communicate with the server in order to report the status of each host or router and receive instructions from the server. This approach is easy to implement and manage, but it increases the load of the network, since it requires additional communications between the managing server and the managed hosts or routers. On the other hand, the distributed control approach distributes the functions of admission control and negotiation in each end host and router. It provides a stronger negotiation function to user. However, it is more difficult to manage and implement, since it requires strong cooperation among hosts and routers.

We use the segment central control approach to perform QoS negotiation and admission control. In this approach, end users can either apply their QoS requirement through the interface for QoS negotiation or choose a default operation to let the QoS management server automatically decide the QoS according to the type of application. In every host and router, there is at least one agent that connects the host or the router with the server. The module relationship among the server, end hosts, and routers is shown in **Fig. 12**.

The server receives the QoS requirements from each managed host, and also receives the current status of resources and their utilization. It then calculates whether the remaining resources can meet the QoS requirement of a new applying media stream. If the remaining resources can not meet the QoS needs of the new application, a QoS renegotiation between existing streams and new applications may occur, provided that the new application has a very high processing priority. Otherwise, the admission control algorithm will reject this application. A management flowchart of the QoS management server is shown in **Fig. 13**.

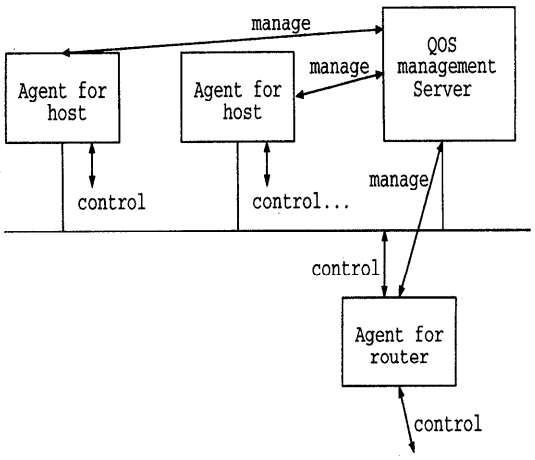


Fig. 12 Module relationship in the segment central control approach.

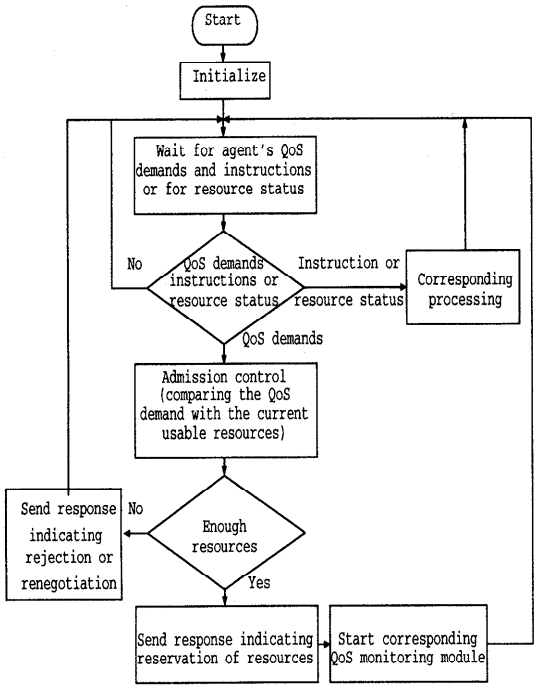


Fig. 13 Management flowchart for QoS negotiation and admission control in the management server.

7. Conclusions

This paper has represented a QoS control mechanism that consists of a framework underlying PTCs (perceptual-time channels) for delivering different media streams according to their characteristics. A PTC is a session between end users that consists of the RSVP protocol and some control algorithms. These algorithms include the QoS definition, packet scheduling, QoS negotiation, and admission control. We mathematically defined the QoS parameters and the implementation formats of these parameters on the basis of the RSVP protocol.

On the basis of the defined QoS parameters and RSVP protocol, we described our packet-scheduling policy and packet-dropping scheme for allocating CPU time and buffers. We also drafted a negotiation and admission control scheme.

Such a mechanism gives rise to a problem as regrads the complexity of QoS parameters, which makes it difficult for users to input or negotiate the required QoS. However, it is hoped that this problem can be solved by using agents and the central control method. Another problem of our approach is the application to a wide area network environment. Though the approach is theoretically useful in a wide-area-network environment, because the RSVP protocol was originally for such an environment, the lack of a scheduling algorithm and admission control scheme for wide area networks and the existence of routers without any QoS control mechanism make QoS control in a wide-area-network environment very difficult. However, our QoS control mechanism works well in a local area network, a segments of a wide area network, or a wide area network where the routers and hosts use the same scheduling policy and admission control scheme based on RSVP.

There are still many other problems demanding further research, such as evaluation of the scheduling algorithm, dropping scheme, and PTCs, measurement of available resources, QoS renegotiation, and management of PTCs. It is nevertheless possible to solve these problems and provide better integrated services on the Internet by using the proposed QoS control mechanism.

Acknowledgments We would like to thank the anonymous referees for their invaluable comments, which significantly helped to improve the quality of the paper.

References

1) Leslie, I.M., et al.: The design and implementation of an operating system to support distributed multimedia applications, *IEEE JSAC.*, Vol.14, No.7, pp.1280–1297 (1996).
2) Newman, P., et al.: IP switching and gigabit routers, *IEEE Communication Magazine*,

pp.92–99 (Jan. 1997).
3) Tennenhouse, D.L., et al.: A survey of active network research, *IEEE Communication Magazine*, pp.80–86 (Jan. 1997).
4) Zhang, Y.X. and Gai, F.: QoS management in the MBone environment, Acta of Electronica Sinica, Vol.23, No.10, pp.32–36 (1995).
5) Chen, H., et al.: Multimedia QoS classification and negotiation manager, *Chinese Journal of Electronics*, Vol.7, No.1, pp.44–48 (1998).
6) Zhang, Y.X., et al.: A study on the router SED-08, *High Tech. Letters*, Vol.7, No.10, pp.32–34 (1997).
7) Ma, H.J. and Zhang, Y.X.: AOSR: An active operating system for routers, *Journal of Computer Research and Development*, to appear (1999).
8) Braden, R., et al.: Integrated services in the Internet architecture: An overview, IETF RFC 1633 (Jul. 1994).
9) Braden, R., et al.: Resource reservation protocol (RSVP), Version 1, functional specification, IETF 2205 (Sep. 1997).
10) Wroclawski, J.: Specification of the controlled-load network element service, IETF RFC 2211 (Sep. 1997).
11) Shenker, S., et al.: Specification of guaranteed quality of service, IETF RFC 2212 (Sep. 1997).
12) Bolliger, J. and Gross, T.: A framework-based approach to the development of network-aware applications, *IEEE Trans. Soft. Eng.*, Vol.24, No.5, pp.376–389 (1998).
13) Eberle, A. and Oertli, E.: Switcherland: A QoS communication architecture for workstation clusters, *Proc. ACM ISCA* '98, Spain (Jun. 1998).
14) Flgd, A. and Fall, K.: Router mechanisms to support end-to-end congestion control, Technical Report, Lawrence Berkeley National Lab., Berkeley (Feb. 1997).
15) Feng, W. and Liu, J.W.-S.: Algorithms for scheduling real-time tasks with input error and end-to-end deadlines, *IEEE Trans. Soft. Eng.*, Vol.23, No.2, pp.93–106 (1997).
16) Mehra, A., et al.: Structuring communication software for quality-of-service guarantees, *IEEE Trans. Soft. Eng.*, Vol.23, No.10, pp.616–635 (1997).
17) Barzialai, T.P., et al.: Design and implementation of an RSVP based quality of service architecture for an integrated service Internet, IEEE JSAC, Vol.16, No.3, pp.397–413 (1998).
18) Baidey, M.L., et al.: Pathfinder: A pattern-based packet classifier, *Proc. ACM SIGCOMM*, London, U.K., pp.115–123 (Aug. 1994).
19) Zhang, L., et al.: RSVP: A new resource reservation protocol, *IEEE Network*, pp.8–18 (Sep. 1993).
20) Shenker, S., et al.: A scheduling service model and a scheduling architecture for an integrated services packet network, Working Paper, Xerox PARC (Aug. 1993).
21) Katcher, D., et al.: Engineering and analysis of fixed priority schedulers, *IEEE Trans. Software*, Vol.19, No.9, pp.920–934 (1993).
22) Campbell, A., et al.: Integrated quality of service for multimedia communications, *IEEE INFOCOM* '93, San Francisco (Mar. 1993).
23) ISO: Quality of service framework outline, ISO/IEC/JTC1/SC21/WG1 N1145 (Mar. 1992).
24) Wellman, M.P.: A market-oriented programming environment and its application to distributed multicommodity flow problems, *Journal of AI Research*, Vol.1, No.1 (1993).

(Received October 9, 1998)
(Accepted June 3, 1999)

YaoXue Zhang was born in Hunan, China, on Jan. 5, 1956. He received the B.S. degree in electronic communication from Xidian University, Xian, China, in 1982. In 1986 and 1989, he received the M.S. degree and the Ph.D. degree in computer science from Tohoku University, Japan, respectively. He joined Tsinghua University in 1990 and has been a full professor since 1993, at the department of Computer Science, Tsinghua University, China. He was a visiting scientist in Laboritory of Computer Science MIT, USA, in 1995, and a visiting professor in University of Aizu, Japan, in 1998, respectively. He is currently leading a research group to study network architecture including management and control method of Quality of Services provided by networks, protocol specification, synthesis, verification and implementation method, network interconnection including definitions of interconnection protocol, routing algorithm, design and implementation of routers for network interconnection and protocol conversion method, information system integration, and network application software.

Zixue Cheng received the M.S. and Ph.D. degrees from Tohoku University in 1990 and 1993, respectively. He was an assistant professor from 1993 to 1999 and has been an associate professor since Apr. 1999, in the Department of Computer Software at the University of Aizu. Currently he is working on distributed algorithms, groupware, and protocol synthesis and implementation. Dr. Cheng is a member of IEEE, ACM, IEICE, and IPSJ.

Xiaochun Wang received the M.S. degree in communication engineering from the college of information engineering, Zhenzhou, China, in 1993. He joined Jiangnan Institute of Computing Technology in 1993 and became a engineer in 1995. He is currently a Ph.D. Candidate in computer science and technology from Tsinghua University, and his research areas are scheduling method, and QoS control.

Hua Chen was borned in Fuzhou, China, on Nov. 1, 1972. He received the B.S. degree in computer science and technology from Tsinghua University, Beijing, China in 1994. From then on, he has been a Ph.D. Candidate in computer science and technology from Tsinghua University. He is currently studying scheduling method in computer and QoS control in Internet.

Shoichi Noguchi was born in Tokyo on March 5th, 1930, and received his B.E., M.E., and D.E. degrees in Electrical Communication Engineering from Tohoku University, Sendai. He joined the Research Institute of Electrical Communication at Tohoku University in 1960, and was a professor at the same University from 1971 to 1993. Dr. Noguchi had been Director of the Computer Certer, Tohoku University, from 1984 to 1990 and Drector of Research Certer for Applied Information Science, Tohoku University, from 1990 to 1993. He had been a professor of Faculty of Engineering, Nihon University, from 1993 to 1997. Dr. Noguchi was President of IPSJ (Information Processing Society of Japan) from 1995 to 1997. He is President of the University of Aizu since April 1997. His main area of interests is information science theory and compuer network fundamentals. He has also been active in the area of parallel processing, computer network architecture, and knowledge engineering fundamentals.