The Internet protocol architecture is being extended to support real-time services. The new architecture has introduced two new classes of service, guaranteed and the controlled load classes in addition to the old "best-effort" class. These services are designed to work with multicast as well as unicast. Frameworks have also been proposed for integrating the Integrated services over other link layers such as ATM, PPP and 802 protocols.

### Table of Contents

- **Introduction**
- **Objectives**
- **The Integrated Services Model**
  - Quality of Service Requirements
    - Real Time Applications
    - Elastic Applications
  - Resource Sharing Requirements
  - Packet Dropping
- **Classes of Service**
  - Guaranteed Service Class
    - How is the Guaranteed Service described?
    - Policing in Guaranteed Service
  - Controlled Load Service
    - How is the Controlled Load Service described?
    - How does Controlled Load Service handle bursts?
    - Admission Control for Controlled Load Service
  - The Best Effort Service class
- **How are the service classes requested?**
- **Support of Integrated Services over other link layers**
  - Integrated Services over ATM
    - Mapping Service Classes over ATM
Introduction

So far, the Internet has been following the "best-effort" delivery model. There is no admission control and the network offers no assurance about the delivery of the packets. Most of the applications on the Internet were elastic in nature, in that they tolerated packet delays and packet losses, and hence they could be served by the "best-effort" model. But, there are new applications today on the Internet that do not necessarily follow the "best-effort" model. These new applications such as voice and video applications, require a finite (small!) bound on the end-to-end delay. For instance, for good quality voice, it is necessary for interactive voice applications to have an end-to-end delay of less than 150 ms. Voice packets arriving later than 150 ms will have to be discarded. This is very different from the traditional data applications.

The emerging multimedia applications have very different characteristics and requirements than the data applications. These new real-time applications are typically less elastic and less tolerant of delay variation. Such applications do not perform well under extreme variations in the delay and excessive dropping of packets. If we were to have a highly congested link, it would be difficult to have a good quality voice or video application running. Besides, these applications do not adapt to congestion, and this leaves the adaptive data applications with less bandwidth. In the current Internet model, the real-time applications do not perform adequately and interfere with the data applications and hence, it is necessary to extend the Internet model in order to support the newer applications. The IETF's Integrated Services group is working on specifying an enhanced service model.

In this survey, we present the proposed classes of service in the new Internet model. We start by looking at the objectives and the assumptions of the extended Internet Model.
Objectives of the extended Internet model

The goal of any network should be to maximize the performance of its resident applications, or in other words the network must meet the application's demands. To support newer applications such as voice and video means that the network must cater to the new multimedia applications by guaranteeing what the applications need. If the network provides the simple "best-effort" class, it is difficult for these applications to meet their requirements. What is really required of the network is to provide multiple classes of service that can cater to these applications as well. It is easy to see that having multiple classes of services is definitely better than the simple "best-effort" single class, and maximizes the performance of the applications. The mapping between the service classes and applications must reflect the application requirements. If the network implicitly chooses the service class to which an application must belong (based on some characteristics of the application), then we would be able to support only a limited class of applications. Besides, the interface between networks and applications would no longer be clean and violates the philosophy in the Internet model. Hence, it is necessary to have the applications explicitly request the service they desire, as opposed to the network implicitly choosing the service class. To decide whether to have admission control in the network or not is an important question. If admission control were not present, then there is no way of the applications specifying what they desire and of the network to indicate whether the applications demands can be met. Admission control is required since performance of the real-time applications is maximized by rejecting additional requests for new flows. Hence, the extended Internet Services must cater to delay sensitive applications by providing multiple classes of service and guaranteeing certain parameters by admission control.

The Integrated Services Model

In order to meet the above mentioned objectives, the Integrated Services model was proposed. A service model is embedded within the network service and invoked by applications to define the set of services they can request. This service model is independent of any specific network artifact, and is based solely on fundamental service requirements.

The Integrated Services (IS) model in the Internet Architecture will support real-time as well as non-real time applications. We look at the assumptions underlying the IS model, and then at various components of the Integrated Services model.
Assumptions of the IS model

- Resources must be explicitly managed in order to meet application requirements.
- The service guarantees for a real-time service cannot be achieved without reservations.
- Though one might have applications that can dynamically adapt to changing network conditions, the end-to-end delay must be bounded.
- Statistical sharing between the real-time and non-real time traffic is advantageous over an entirely new parallel infrastructure for real-time traffic. In other words, it is desirable to use the Internet as a common infrastructure for both non-real-time and real-time communication.

Quality of Service Requirements

The core service model is almost exclusively concerned with the time-delivery of packets. Thus, per-packet delay is the central quantity about which the network makes quality of service commitments. The only quantity about which quantitative service requirements are made are bounds on the maximum and the minimum delays [RFC1633].

The degree to which application performance depends on low delay service varies widely. Real Time applications are the class of applications that needs the data in each packet by a certain time, and if the data has not arrived by then, the data is essentially worthless. On the other hand, the elastic applications will always wait for the data to arrive.

Real-Time Applications

The figures below show the nature of the real-time applications and the delay adaptive applications.
Figure 1: Nature of Hard Real Time Applications (from [SHENKER95])

Figure 2: Delay Adaptive (Controlled Load App.)
Though real-time applications will fall along a continuum, we can broadly subdivide the class of real-time applications into two groups, one consisting of applications that are intolerant to delay, and the tolerant or delay-adaptive real-time applications. The intolerant applications will have good performance only if they are given a perfectly reliable upper bound on delay (see *figure 1*). Hence, in order to support these hard real-time applications, the network must guarantee a perfectly reliable upper bound on the delay. There are other applications that do not require a perfectly reliable delay bound. These applications can probably adjust to variations in the delay, and these applications could hence be called the adaptive applications (see *figure 2*). Based on these applications, we can define two kinds of real-time service classes: guaranteed service and controlled load service. The guaranteed service gives a perfectly reliable upper bound on the delays, and the controlled load service supplies a fairly reliable, but not perfectly reliable, delay bound. The guaranteed service is based on the worst case assumptions on the behaviour of the other flows while the controlled load service bounds could be computed with proper conservative predictions about the behaviour of other flows. The controlled load service does not try to minimize the delay of every packet, but rather pulls in the tail of the delay distribution.

### Elastic Applications

Elastic applications always wait for data to arrive. The key characteristic of these applications is that they typically use the arriving data immediately, rather than buffering it for later, and will always choose to wait for the incoming data rather than proceed without it. Typically, for a given delay distribution, the perceived performance of elastic applications will depend more on the average delay than on the tail of the delay distribution. An appropriate service model for the elastic applications is the existing *best-effort* service already existing in the Internet. It is also important to note that these applications are not subject to admission control. *Figure 3* shows that the network can be maximally utilized by supporting the elastic applications.
Thus, based on the quality of service commitments, we can introduce two real-time service classes, the **guaranteed** and the **controlled load** class, in addition to the best-effort class that exists.

## Resource Sharing Requirements

The quality of service commitments dictate how the network must allocate its resources among the individual flows. This allocation is typically negotiated on a flow-by-flow basis as each flow requests admission to the network. It does not address any of the policy issues that arise when one looks at collections of flows. These policy issues are addressed by resource-sharing service commitments. While delay was the primary (only) quantity of interest for individual quality of service, the quantity of primary interest in resource-sharing is *aggregate bandwidth*. This component of the service model, *link-sharing*, is concerned with the sharing of the aggregate bandwidth of a link among various collective entities according to some set of specific shares.

An idealized fluid model of instantaneous link-sharing with proportional sharing of excess is the fluid processor sharing model [PAREKH92]. In this model, the available bandwidth is shared between the packets in the queue in proportion to the assigned shares of the resource. Since the fluid model is an unrealistic idealization, the goal of the actual service model should be to approximate, as close as possible, the bandwidth shares produced by the ideal fluid model.
Packet Dropping

Within a flow, all the packets are not necessarily of equal importance. For example, in an MPEG stream, the I frames are more important than the P or B frames. Losing a P or a B frame is preferable over losing an I frame. Hence, the service model has a *preemptable* packet service, whereby some of the packets could be marked as preemptable. The network could either discard these preemptable packets and try to meet the some of its quantitative service commitments.

Classes of Service

Based on the above characteristics of the service model, three service classes are being proposed: the Guaranteed and the Controlled Load Service for the real-time applications and the best-effort service for the applications that are delay-tolerant. A **TSpec** and a **RSpec** specify a service class completely. The TSpec describes the flow's traffic characteristics to the network. The RSpec is used by the flow to request a specific requirement from the network. This requirement could be in terms of a bounded delay or something else. We will be using these two definitions in the sections below.

Guaranteed Service Class

The Guaranteed service class provides firm end-to-end delay guarantees. This service guarantees both delay and bandwidth. A flow is described using a **token bucket**. Using this traffic description, the network can compute various parameters describing how it will handle the flow, and by combining the parameters, it is possible to compute the maximum delay a packet can experience. The delay has two parts, a fixed delay and a queuing delay. The fixed delay is a function of the chosen path, comprising of transmission delays etc., and is determined by the setup mechanism. The queuing delay, determined by guaranteed service, is primarily a function of two parameters: the token bucket, and the data rate the application requests. The end-to-end behaviour provided conforms to the **fluid** model, in that the delivered queuing delays do not exceed the fluid delays by more than a specified error bound.

The Guaranteed service does not control the minimum or average delay of datagrams, merely the maximal queuing delay. The service guarantees that datagrams will arrive within the requested delivery time and will not be discarded due to queue overflows, provided the flow's traffic stays within its specified traffic parameters. This service is intended for applications which need a firm guarantee that a datagram will arrive no later than a certain delay bound. An important aspect to note here is that the service does not attempt to minimize the delay **jitter**, but merely controls the maximal queuing delay. Since the guaranteed delay bound is a firm one, the delay has to be set large enough to cover extremely
rare cases of long queueing delays.

How is the Guaranteed Service described?

Guaranteed Service is described using both a TSpec and an RSpec. The TSpec here is a token bucket\((r,b)\), a peak rate \((p)\), a minimum policed unit \((m)\), and a maximum datagram size \((M)\). The token bucket has a bucket depth, \(b\) and a bucket rate, \(r\). The peak rate \(p\), is the maximum rate at which the source can inject bursts of traffic. A minimum policed unit of size \(m\), means that any datagram of size less than \(m\) will be counted as being of size \(m\). The RSpec is described using a rate \(R\) and a slack term, \(S\) in microseconds. The slack term signifies the difference between the desired delay and the delay obtained by using a reservation rate \(R\). Using the slack term, one can request a specific delay guarantee.

Policing in Guaranteed Service

There are two forms of policing, *simple policing* and *reshaping*. Comparing arriving traffic against a TSpec is simple policing, while reshaping refers to reconstruction of a traffic's shape to conform to the TSpec. A flow violates the TSpec if the reshaping fails. The reshaping fails when the reshaping buffer overflows.

Policing is only done at the edge of the network. Reshaping is done at all heterogenous source branch points and at all source merge points. A heterogenous source branch point is a point where the multicast distribution tree from a source branches to multiple distinct paths, and the TSpecs' of the reservations on the various outgoing links are not all the same. Reshaping needs to be done only if the TSpec on the outgoing link is less than the TSpec reserved on the immediate upstream link.

Controlled Load Service

*Controlled load Service* is an enhanced quality of service intended to support applications requiring better performance than that which would be provided by the traditional best-effort service. It approximates the end-to-end behaviour provided to best effort service under unloaded conditions. The assumption here is that under unloaded conditions, a very high percentage of the transmitted packets are successfully delivered to the end-nodes, and the transmission delay experienced by a very high percentage of the delivered packets will not vary much from the minimum transit delay.

The controlled load service is particularly aimed at supporting the "predictive" or the "delay-adaptive real-time" class discussed earlier. The delay-adaptive class works well on unloaded networks, but degrades quickly under overloaded conditions.

The network ensures that adequate bandwidth and packet processing resources are available to handle the requested level of traffic. The controlled load service does not make use of specific target values for control parameters such as delay or loss. Acceptance of a controlled-load request is merely a
commitment to provide the flow with a service closely equivalent to that provided to uncontrolled traffic under lightly loaded conditions. Over all timescales significantly larger than the burst time, a controlled load service flow may experience little or no average packet queueing delay, and little or no congestion loss.

How is the Controlled Load Service described?

The controlled load service is described only using a TSpec. Since the network does not give any quantitative guarantees, there is no RSpec required. A Controlled Load flow is described by a token bucket \((r,b)\), a peak rate \((p)\), a minimum policed unit \((m)\) and a maximum packet size \((M)\). This is similar to the way the guaranteed service was described. The only difference here is that the network does not promise any delay guarantees. Token bucket description of Controlled Load Service includes a bucket rate \(r\), and a bucket depth \(b\). Over all time periods \(T\), length of the burst should never exceed \(rT+b\). Packets not conforming to this, or violating the MTU size, are considered non-conformant. The controlled load flows not experiencing excess traffic will get the contracted quality of service, and the network elements will prevent excess controlled load traffic from unfairly impacting the handling of arriving best-effort traffic. The excess traffic will be forwarded on a best-effort basis.

How does Controlled Service handle bursts?

It is possible that if a flow receives exactly its promised token rate at all times, queuing caused by an over-rate burst may never clear. This will lead to a permanent increase in the queueing delay. This situation can occur if a flow continues to generate traffic at exactly the token rate after emitting the burst. To control the long-term burst traffic effects, the controlled load service has several options. It can borrow bandwidth needed to clear bursts from the network, using an explicit borrowing scheme within the traffic scheduler or an implicit scheme based on statistical multiplexing and measurement based admission control. The buffer management is flexible enough to make more buffering available to handle additional bursts. The current documents do not specify any specific way to handle the controlled load service. It places the onus of meeting the requirements on the admission control algorithm. Information from measurement of the aggregate traffic flow or specific knowledge of traffic statistics could be used by the admission control algorithm to have a multiplexing gain.

Admission Control for Controlled Load Service

As discussed in the above section, the controlled load service does not make any guarantees about the delay but promises to approximate the end-to-end behaviour of the applications to the best-effort applications under low-load. The admission control algorithm has to make sure that this approximation is met. The admission control algorithm can use measurements of the aggregate traffic flow or specific information about the traffic. A couple of schemes are being proposed for measurement-based admission control. These schemes take admission control decisions based on the measurements of the current aggregate network load. The advantage of doing this is that one can fully utilize the network resources. However, the disadvantage is that the admission control algorithm is difficult to design and justify since it is based on measurement-data instead of the worst-case characterization of the traffic.
The Best Effort Service class

The current model that is followed in the Internet is the best-effort model. Most data applications are most suited to operating in this way. The applications like telnet, ftp etc. do not have any hard deadlines in terms of delay. They wait for data and process it as soon as they receive it. This service class will continue to be supported in addition to the real-time classes. Applications can choose to use this service class, and if they find the delay unacceptable, can use one of the other service classes. The Best Effort Service class does not have a TSpec or an RSpec. There are no guarantees by the network whatsoever. The network does not do any admission control as well for this class.

How are the service classes requested?

The integrated services model provides the ability for applications to choose among multiple classes of service. This means that all network nodes, such as the routers along the path of a flow, must be informed of the requested class of service and the respective parameters. The latter mechanism can be implemented by use of a resource reservation protocol, such as RSVP, the Resource Reservation Protocol.

The design of RSVP lends itself to be used with a variety of QoS control services. RSVP specification does not define the internal format of the RSVP protocol fields, or objects. It treats these objects as opaque and deals only with the setup mechanism. RSVP was designed to support both unicast and multicast applications. RSVP supports heterogenous QoS. Heterogenous QoS means that different receivers in the same multicast group can request different QoS. This heterogeneity allows some receivers to have reservations while there could be others receiving the same traffic using the best-effort service. For a good overview of the RSVP protocol, refer to [DRAFT-RSVP].
Support of Integrated Services over specific link layers

IP is interoperable today over several different link layers (with respect to IP). These link layers include ATM, 802 technologies such as the Ethernet, point to point links such as PPP. Some of these link layers, such as ATM already have QoS abilities. In order for the newly proposed service classes to be useful to the end-user, it is necessary to interoperate the Integrated Services with the other technologies. Efforts are under way to support the Integrated Services Model and the Resource Reservation Protocol (RSVP) over the different link layers. This section discusses the support of Integrated Services over ATM, 802 technologies and low-bitrate links.

Integrated Services over ATM

Today, unicast IP using best effort can be supported over ATM [RFC1577]. IP multicasting can be done over ATM using MARS [MARS], the Multicast Address Resolution Server. The ATM Forum's LAN Emulation (LANE) and Multiprotocol over ATM (MPOA) also address the issue of IP best effort traffic over ATM. ATM supports various classes of service, and so it should be possible to interoperate easily with ATM. In order to support the Internet Integrated Services (IIS) over ATM, the following key issues must be addressed.

- Mapping Service Classes
- Supporting RSVP over ATM

We address each of these briefly here.

Mapping Service Classes

ATM supports several classes of service.

1. CBR - Constant Bit Rate
2. rt-VBR - Real-time Variable Bit Rate
3. nrt-VBR - Non Real-time Variable Bit Rate
4. ABR - Available Bit Rate
5. UBR - Unspecified Bit Rate

The Internet service classes: guaranteed, controlled load and best-effort must be mapped into the most appropriate ATM service class. Best effort service is appropriately supported using the UBR class. The guaranteed service could be supported using CBR or rt-VBR, while we have two choices for controlled load service, nrt-VBR and ABR.

<table>
<thead>
<tr>
<th>IP Service Class</th>
<th>ATM Service Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guaranteed Service</td>
<td>CBR or rt-VBR</td>
</tr>
</tbody>
</table>
Controlled Load Service | CBR, *nrt-VBR* or ABR  
Best-Effort Service | *UBR*, any ATM Class  

**Table 1**: *Mapping of Internet Integrated service classes to ATM service classes.*

Table 1 shows the possible mappings between the Integrated Service Classes and the ATM Service Classes. The ATM Service classes best suitable for the corresponding IP Service Class are shown italicised in the table. The *rt-VBR* class is more appropriate for supporting Guaranteed Service since it can utilize allocated bandwidth left unused by a source and support more bursty sources. The *nrt-VBR* is good for Controlled Load Service since it provides for allocation of bandwidth and buffers. *UBR* is an obvious choice for the Best-Effort Service. [DRAFT-ATM-MAPPING] addresses all the issues related to mapping the IP classes over the ATM Classes.

**Supporting RSVP over ATM**

RSVP is the Internet signalling protocol used to set up flow reservations. The RSVP messages must follow the same IP path throughout the network. We tabulate some of the options that we have to support RSVP over ATM.

<table>
<thead>
<tr>
<th>ATM Option</th>
<th>Supporting QoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PVC</td>
<td>QoS of IP should be equivalent to QoS of PVC.</td>
</tr>
<tr>
<td>SVC</td>
<td>If single SVC is used per RSVP flow, then the total number of flows are restricted. It has problems of scalability.</td>
</tr>
<tr>
<td>Point to Multipoint</td>
<td>RSVP supports heterogenous receivers, but ATM does not.</td>
</tr>
<tr>
<td>Multicast Servers</td>
<td>Using Multicast servers will allow heterogenous receivers, since it can interpret RSVP packets. But it will have scalability issues.</td>
</tr>
</tbody>
</table>

**Table 2**: *ATM Options for supporting RSVP.*

The key issue in supporting RSVP over ATM is to make sure when a Virtual Circuit is going to be setup for a specific Quality of Service related to RSVP. The scalability issue is how many VC's should be created to support the RSVP Flows. A single VC per flow does not scale very well, since we have limited number of VCs. RSVP supports heterogenous receivers, and this is extremely difficult to support in the ATM world, since a point to multicast supports only homogenous receivers (with same QoS).
**Integrated Services over 802 LANs**

Most of the IP traffic today goes over 802 Lans such as the Ethernet. It is essential to create a framework inorder to provide the integrated service classes over the 802 Lans. Traditionally, the 802 technologies have been required to provide only the best-effort service. The Ethernet/802.3 LANs do not support any mechanism of providing quality of service, but the Token Ring and FDDI have support for traffic classes. The recent 802.1p standard aims at supporting expedited traffic classes and dynamic multicast filtering in the 802 LANS. However, supporting traffic classes over Ethernet would be restricted to the host. Shared Ethernet can at best offer loose statistical service guarantees because CSMA/CD is non-deterministic. But it will be possible to have better guarantees on token ring and FDDI, as they support Frame Priority, Reserved Priority and Access Priority. Depending on the basic topology of the LAN and the ability to support traffic classes, six scenarios are possible

<table>
<thead>
<tr>
<th>LAN Topology</th>
<th>Description</th>
<th>Kind of guarantees possible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Topology without traffic classes</td>
<td>Ethernet/802.3 without 802.1p. No isolation between flows possible.</td>
<td>No Guarantees Possible</td>
</tr>
<tr>
<td>Shared Topology with traffic classes</td>
<td>Token Ring/FDDI/Ethernet with 802.1p.</td>
<td>Only <em>statistical guarantees</em> for shared ethernet, but <em>better guarantees</em> for Token Ring and FDDI</td>
</tr>
<tr>
<td>Switched half duplex without traffic classes</td>
<td>Ethernet/802.3 without 802.1p.</td>
<td>No isolation and hence no guarantees.</td>
</tr>
<tr>
<td>Switched half duplex with traffic classes</td>
<td>Token Ring/FDDI/Ethernet with 802.1p. Only two senders in contention.</td>
<td>Better statistical guarantees than shared on ethernet and better guarantees for Token Ring and FDDI.</td>
</tr>
<tr>
<td>Switched full duplex without traffic classes</td>
<td>switched Ethernet and token ring. (again, no isolation possible in absence of traffic classes)</td>
<td>Again, no guarantees possible.</td>
</tr>
</tbody>
</table>
Switched full duplex with traffic classes | Most capable for flow separation and scheduling. | Guarantees possible. Better than the rest.

**Table 3: LAN Topologies**

To support the integrated services framework, an entity called the Bandwidth Manager (BM) has been proposed. BM could be divided into two parts, the Requestor and the Allocator. The Requestor resides in every host and provides an interface between protocols such as RSVP, STII, SNMP etc. to initiate reservation in the link layer domain. The Bandwidth Allocator could be distributed or centralized and is left open to implementation. The Bandwidth Allocator is responsible for admission control and maintaining state about the allocation of resources in the subnet. It is also responsible for labelling flows. For instance, the BM can tag packets belonging to a particular flow with appropriate priorities which maps them to the correct traffic class. Refer to [DRAFT-LAN-MAPPING] for details.

### Integrated Services over Point-to-Point Links

Today, connectivity using IP makes use of low-bit rate links such as 14.4/28.8 and 56 kbps modems, 64 kbps ISDN channels as well as sub-T1 channels. As of today, these links are not appropriate for supporting real-time applications with stringent delay bounds. Consider a 1500 byte packet being transferred over a 28.8 kbps modem link. The transfer makes the link unavailable to other applications for about 400 ms. Clearly, any delay stringent application will not be able to meet its requirements under such conditions. This calls for newer encapsulation formats. In addition to this, the header overhead incurred on such links outshadows the real-time information. For instance, typical audio payloads are about 20 bytes and the header overhead is atleast 44 bytes. This calls for newer header compressibility techniques.

### Real-Time Encapsulation

The end-to-end delay on a point-to-point link needs to be bounded for real-time applications. This means that other transfers on the point-to-point link should be preemtptable. In other words, we should be able to suspend and resume or abort an ongoing best-effort flow to support the transfer of a real-time flow. This can be achieved by *real-time encapsulation*. The goal of real-time encapsulation is to minimize the delay. Several techniques exist to minimize delay such as

- **Using IP fragmentation** - We could use IP fragmentation to reduce the fragment size transferred. However, this is going to increase the overhead for the transfer. Increase the fragment size would incur less header overhead, but increased delay.

- **Using link layer mechanisms** - Capabilities of the hardware and the lower layer can be utilized to allow for preemption of an ongoing transfer. If the hardware supports byte-by-byte control, then we can have a desired mechanism to fine control the delay. This increases the overhead only when required. If the hardware is frame-oriented, there is a tradeoff between the overheads from frame size and the variation in access delay desired, since reducing the frame size increases the overhead.
Summary

The proposed Integrated Services model was discussed. The Internet needs to support the real-time applications in addition to the elastic applications. In order to support the real-time applications, delay guarantees must be provided by the network. The hard-real time applications require a firm maximum bound. However, there are also adaptive real-time applications that can do without a firm bound. The guaranteed and the controlled load service aim at supporting the hard real-time and the adaptive real-time applications respectively. In order to accomplish this, each flow needs to reserve bandwidth as well as indicate its desired quality of service to all nodes (routers) along its path. This will be done using the Reservation Setup Protocol, RSVP. Since IP is interoperable over multiple sub-net technologies today, these new services should also be interoperable over these technologies. Some of these subnets have capabilities to support reservation and quality of service, while others do not. The Integrated Services over Specific Link Layers group is working on supporting the Integrated Services over ATM, 802 LANs and point-to-point links.

In the future, it will be possible to support real-time applications over the Internet. Considerable work is being done towards admission control for the adaptive real-time class. The delay adaptive applications will help to maximally utilize the network resources and still be real-time. In future, we might have more delay-adaptive applications coming up that can make use of the new service.

References


   IETF draft - specifies guaranteed QoS

   IETF Draft

   IETF Draft - Controlled Load Service

   IETF Draft - Supporting RSVP over ATM

    IETF Draft - Specifies several ways of mapping CLS and GS with ATM.

    IETF Draft - Mapping Integrated Services over 802 Lans

    Classical paper: proves that maximum delay bounded with token bucket and WFQ

    IETF Draft - RSVP

    IETF RFC: supporting Classical IP over ATM

    Excellent book - covers various issues on Gigabit Networking.
Appendix

Definitions

- **Quality of Service** refers to the nature of the packet delivery service provided, as described by parameters such as achieved bandwidth, packet delay, and packet loss rates.
- **Network Element** is any component of an internetwork which directly handles data packets and thus is potentially capable of exercising QoS control over data flowing through it.
- **Flow** a set of packets traversing a network element all of which are covered by the same request for control of quality of service.
- **TSpec** is the *traffic specification* described by the applications to the network.
- **RSpec** is the *Service request specification* by the flow to the network of the quality of service desired.
- **Burst Time** refers to the time required for the flow's maximum size data burst to be transmitted at the flow's requested transmission rate.
- **Jitter** is the difference between the minimal and maximal datagram delays.

2. **Token Bucket**

![Token Bucket Diagram]

*Figure 4: Token Bucket*
The *Token Bucket* is a traffic specification consisting of a "token rate" $r$ and a "bucket size" $b$. Here $r$ specifies the continually sustainable data rate, while $b$ specifies the extent to which the data rate can exceed the sustained rate for short periods of time. The rate, $r$, denotes the rate at which the tokens are placed in the bucket (as shown in [figure 4](#)). If the bucket becomes full, the excess tokens are discarded. An arriving packet is placed in a receive buffer, as shown in the figure. To transmit the packet, tokens are removed from the bucket equal to the size of the packet. The traffic obeys the rule that over all time periods of length $T$, the amount of data sent cannot exceed $rT + b$. This is, however, difficult to police. For a detailed discussion of the token bucket, and other traffic shaping schemes, see [PARTRIDGE](http://www.cis.ohio-state.edu/~jain/cis788-97/integrated_services/index.htm).