QoS Management at the Transport Layer

V. Tsoucasidis and S. Wei

Department of Computer Science
SUNY at Stony Brook
Stony Brook, New York 11794-4400

Abstract
This work reports on further implementation and testing of the Application Oriented Transport Protocol (AOTP) [9, 11]. AOTP is an experimental protocol above IP that provides end-to-end transport service with functionality to trade off Reliability, Throughput and Jitter in order to support the Application Layer with the required Quality of Service. Our protocol, based on QoS tradeoffs, can favor application- and/or user-specific QoS characteristics of importance, at the expense of others, less significant. In this paper we outline the protocol's specification and mechanisms. AOTP provides adjustable partially reliable service, priority-based error recovery strategy and dynamic playback management. We have tested the mechanisms using a Video Transmission application and a simulated, low-bandwidth environment with variable error characteristics. We characterize the tradeoffs associated with these mechanisms; we argue that the application layer can yield significant QoS achievements even within the confines of given environmental constraints, when the transport layer utilizes efficiently the tradeoffs developed in the communication system.

1. Introduction
Packet-switched networks possess a significant advantage in today's data communications: better utilization of resources. Using packets of variable length, and connections that accommodate several application flows at a time, packet-based communications can dominate the market, since they can yield lower communication costs for vendors and customers. However, due to its accommodating multifarious applications designed to take advantage of its services, today's Internet experiences a major weakness in quality. In this work we assume that the major parameters that can be used to determine an application's request for Quality of Service are Bandwidth, Delay, Delay Variation (Jitter), and Reliability. Therefore, Quality of Service (QoS) Management becomes an application-specific process. That is, applications can be classified according to their distinct characteristic(s) of importance. For example, some applications can tolerate packet losses but are sensitive to delays and jitter, while others are time-tolerant but require data integrity. In addition, QoS Management becomes a user-oriented process; each of the above-mentioned characteristics of QoS entails dependencies from the others; favoring the one characteristic might have a significant impact on another. In such occasions, the user should be able to decide as to the characteristic of importance.

Several mechanisms have been proposed to augment Quality of Service in order to enable real-time multimedia applications over IP. Most of these mechanisms provide solutions to reserve resources, request Bandwidth on Demand (BoD), or to support differentiated services, proposing modifications for IP-level devices. However, Quality of Service (QoS) can be managed at several layers of the network architecture, from the physical to the application. In this work we aspire to application Quality of Service considering the tradeoffs developed at the error recovery mechanisms of the transport layer, since, in most cases, reliability can be achieved using a retransmission mechanism, thus affecting Throughput, Delay and Jitter as well.

2. Real-time Transport Service
The evolution of multimedia applications over the Internet during the last years has raised the demand for a new service class of transport protocols. The TCP/IP stack, having the traditional TCP reliable/ordered service or UDP unreliable/unordered service is not sufficient for a wide range of applications. Application-oriented protocols that enable/manage QoS tradeoffs, can offer an alternative solution. For example, since video or audio streams can tolerate packet losses, it could be the user's choice to favor bandwidth at the expense of reliability.

There are recently several research papers discussing the new requirements at the transport level to satisfy multimedia or other applications that require high bandwidth, trading QoS characteristics that are not significant, and favoring others of greater importance. At this layer, retransmission of lost or "corrupted" packets is a mechanism widely used to ensure reliability. This mechanism however, increases the number of Round Trip Times (RTTs) required for message delivery, therefore increasing delay and jitter which are crucial for most multimedia communications (e.g. video transmission).
Partial reliability - partial retransmission is one solution as proposed at [1, 6, 7, 8] emphasizing on a sender-based decision as to the need for retransmission. Dempsey in [3] supports the use of retransmission as effective way to handle loss of packets even for continuous media applications. This approach increases delay, which is crucial for real-time multimedia, especially for high-bandwidth links. The use of Forward Error Correction (FEC) mechanism eliminates the ability for flexible tradeoffs, but reduces the number of required RTTs. However, it could be considered as a waste of bandwidth, in case an application could tolerate packet losses or the underlying link was reliable. For the case of congestion, the use of FEC cannot contribute a significant enhancement; packets will be dropped and retransmission will take place. Recently, the XTP Forum [14] has developed a set of protocols that offer high-throughput compared to TCP and even UDP especially over high-bandwidth links. The XTP stack is flexible and widely portable; it can adapt to application specific requirements setting end-to-end unnecessary features, and according to the reported results it can replace the transport protocols of traditional networks, although it aims mainly at high-bandwidth links. Kleinrock in [5] describes the latency/bandwidth tradeoff for such networks, in detail. Reference [12] presents the reliability/goodput tradeoff also as a function of energy (in the form of transmission effort) in the Wave and Wart Protocol.

The Real Time Transport Protocol (RTP) is another protocol under research [2, 10] widely used for audio and video-conference or other multi-participant real-time applications. RTP uses the services of an end-to-end transport protocol, such as end-to-end delivery, framing, multiplexing, and also multicasting. An IP-level protocol, the Internet Stream Protocol (RTPC 1190), provides end-to-end guarantees but requires routers to maintain state information describing the streams of packets flowing through them.

The authors in [9] have proposed avoidance of redundant data to facilitate error recovery when the bandwidth is low, the application can tolerate losses, and/or the link is reliable, taking the risk of retransmission (hence additional RTTs) when the level of minimum reliability requirement has been overreached. The protocol presented here is further refined to trade reliability for throughput and/or minor jitter. AOTP implements this tradeoff efficiently using a retransmission policy based on the significance of the dropped frames and the reliability of jitter requirement. The level of the trade can be adjusted according to the available bandwidth and the reliability requirements of the user, favoring one characteristic at the expense of another (e.g., image quality for speed). Using a priority service, AOTP achieves maximum throughput and/or minor jitter at low cost to image quality.


In AOTP, the receiver decides as to the need for retransmission. This design facilitates dynamic behavior of the protocol, since the reliability level might change at runtime; then the receiver should ask for packets based on the calculated receiving rate and the desired level of reliability. This situation can be exploited using a partially reliable service with priority control, where a mechanism to drop frames of lower priority (e.g., P frames in MPEG) minimizes the cost of missing frames to image quality. Therefore a main concern of AOTP is to achieve maximum throughput and maintain the required level of reliability and/or jitter.

Application level reliability does not necessarily reflect the reliability level at the transport layer. In MPEG for example, bi-directional frames are of no use to the application layer when the associated intrapicture frames are missing. Hence, the number of frames delivered at the receiver's end is not equal to the number of frames "played" at the application layer. Requesting retransmission for low-priority frames unnecessarily degrades throughput and yields no improvement for the application-level QoS parameters. Similarly, a playback buffer that holds such low-priority frames, might not leave available space for the significant intrapicture frames; this dynamic can introduce jitter, additional delays for the in-synchronous new frames delivered, and even packet drops when the buffer is overflowed. On the other hand, jitter tolerance can be made to improve the level of reliability since the delayed packets can still be useful at the application layer; and tolerance in reliability can improve throughput and balance jitter significantly.

AOTP aims at exploiting these tradeoffs in an application oriented manner. That is, to use advantage of the application specific characteristics and with respect to network constraints (i.e. bandwidth, latency) and QoS requirements, to adjust appropriately the sender's transmission rate, error control, and recovery strategy. In order to achieve this, AOTP offers to transport services: an adjustable partially reliable service with priority control, and a best effort service.

3.1 Specification

Our current design calls for a 6-byte header. Functionality for synchronization is not presently supported; RTP is proved to be an efficient protocol to provide this function and can be used in conjunction with AOTP and without functionality overlap. The protocol's header is associated with the mechanisms provided, and as stated earlier, these mechanisms are under further investigation and development. AOTP's header is presently composed of the following fields:
3.2 Implementation.

As shown in figure, AOTP combines Negative Acknowledgments (NACKs) and Selective Acknowledgments (SACKs) to enable a receiver-based policy for timely retransmission. There is a difference between receiver-based and sender-based decision. When a SACK is sent reporting the delivered packets, the sender can retransmit all the missing ones or decide which ones to select. In case of NACKs, the receiver is the one deciding which ones to report. Hence, the user can decide whether to favor, for example, application quality or speed and dynamically adjust these parameters to new requirements while the application is running.

3.2.1 Partially Reliable Service

For the Partially reliable service, the high-level protocol can set the receiving percentage. No matter what the percentage is, the significant frames (i.e., the I-frame) should be received before delivering other video frames.

In both service categories discussed below, the receiver sends an accumulated acknowledgment to the sender so that the sender can clear its buffer.

For a best effort partially reliable service, the sender keeps the pipe full, initially without worrying of packet loss or reordering. The receiver adapts the timeout as appropriate to handle the delayed packets, and the sender is informed of dropped packets using NACKs. However, the decision is taken by the receiver since, according to our algorithm, the receiver computes the receiving rate, and when the user constraint is violated, it sends back NACKs. This approach reduces the NACKs reports since not all dropped packets are reported, but instead, only the ones necessary to bring back the packet loss to an affordable rate. At this session the protocol does not use a priority mechanism as described below.

![Figure 1: AOTP peer interactions](image)

As shown in Figure 1, AOTP uses receiver-based packet loss recovery mechanism. The application can set

---

**Table 1: AOTP Frame Format**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port</td>
<td>Source access-point for the lower-level protocol.</td>
</tr>
<tr>
<td>Destination Port</td>
<td>Destination access-point for the lower-level protocol.</td>
</tr>
<tr>
<td>Sequence Number</td>
<td>Sequence number for data segments.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of data segment: DATA, SACK, NACK, ACK.</td>
</tr>
<tr>
<td>Priority</td>
<td>Priority level of the frame.</td>
</tr>
<tr>
<td>Stream Control</td>
<td>Stream identifier for data segment.</td>
</tr>
<tr>
<td>Payload</td>
<td>Payload of the data segment.</td>
</tr>
<tr>
<td>Checksum</td>
<td>Checksum of the data segment.</td>
</tr>
</tbody>
</table>
the desired (or minimum) percentage of the message it wishes to receive. The sender fragments the packets when their size exceeds the MTU. Each stream is identified within time intervals by a unique ID, and each frame is assigned with a sequence number. The sender stores a copy for every frame it sends. The receiver starts a timer upon receipt of the first frame. When the timer expires, it checks the percentage received.

```
if RP < DP
   send NACKs
else
   reassemble
```

If the Receiving Percentage (RP) is less than the Desired Percentage (DP) as it is set by the application, the sender sends NACKs for those frames which have not been received yet. Otherwise, the receiver reassembles the frames and sends a SACK (Selective Acknowledgment) to the sender in order to clear the sender’s buffer without worrying about the significance of the missing frames. The sender, upon receipt of a NACK, retransmits the reported lost frame.

The algorithm used for partially reliable service with priority control is described below. RR is the Receiving Rate, ER is the Expected Rate which can be calculated easily since the sending rate is constant, AR is the Accepted Rate described by the receiver, and NR is the new rate calculated after the retransmission of NACKed frames. Since the mechanism targets real-time applications, the measurements are taken using time-intervals or a constant number of fixed size packets rather than marked fragments. The use of time intervals to trigger measurements enables the receiver to reflect the reliability/quality requirement and keep a timely operation according to the real-time constraints of its application. Adjustment of reliability cannot be achieved using a mechanism that measures time needed for a complete delivery of a number of frames. This approach could work in case the user specifies preferences using time metrics. The I, B, P frames reported below, represent different priority levels and are explained further in the next section.

```
for (Time interval = T

if (RR < ER)
   send NACK with I, B, P frames to satisfy: RR >= AR
   where BER=RR-retransmitted (I+B+P) frames;
else
   NACK only missing I-Frames;
   deliver to higher layers
```

4. Results and Discussion.

AOTP is implemented on the x-kernel platform [13] simulating a fairly low bandwidth environment. We have tested the retransmission mechanism using two x-kernel protocols VDROP and VDELAY, which are virtual protocols that are combined to simulate network conditions. VDROP drops packets at a certain regular interval and VDELAY holds (delays) the packet for a certain amount of time, with the interval and time being set by the high level protocol.

The sender sends a video stream and the receiver calculates the receiving rate. When the requirements are met, the receiver passes the stream up to the receiver application. For the testing protocol configured above AOTP we have used the MPEG System [2, 4] as reference for system coding. This system specifies the system multiplexer and encoder to produce a video and audio stream with a system reference clock as a base. MPEG compresses the video and audio streams at about 1.5 Mbps per channel. It concentrates on reducing the temporal redundancy across the frames and there are three types of pictures that consider: Intra-pictures (I-Frames), Predicted pictures (P-Frames), and Bi-directional Frames (B-Frames). An I-Frame is the JPEG version of the corresponding frame in the video source; B and P-Frames are not self contained - they specify relative differences from a reference frame which is the I-Frame for the P-Frame, and I and P-Frames for the B-Frame. The video is encoded and encapsulated in the network protocol and sent across the net. In MPEG, the highest degree of compression is achieved by using as many B frames as possible. Note by using B frames the order of the images in a MPEG-coded data stream often differs from the actual decoding order. A P frame is decoded before the B frame and displayed after it. Decompression of the B frame requires the P frame’s data, which means that the receiver must buffer the sequence of the related frames before delivering them to the application protocol for decoding the image. Therefore, we do not send the frames in the order they are produced, but instead, we reorder them at the sender as 1P B B B B instead of 1B B B B B I and we prioritize them accordingly.

The table (see appendix) shows the average time it takes to deliver 1MB over a low-bandwidth environment. Measurements were taken every 1024 packets received; For example, as shown in the 2nd test with 10% Dropping Rate and 100% Reliability, it takes 22 seconds to receive the first 1024 packets (each packet has a size of 1KB, so the total size is 1MB). It takes another 22 seconds to receive 1024 packets (so the total packets received is 2048), and so on. This results in 22sec average time for 1MB stream with a 100KB buffer size. For the testing we have used the receiver’s clock and set the intervals at the receiver to avoid synchronization problems of the two hosts where the client and server were running. Each test was taken several
times the average is reported. Note that the throughput reported here is calculated as Bytes Received / Time ("goodput").

In a routed network, IP packets usually arrive out of sequence since the take different routes to their destinations, and/or experience delay variation, Jitter is the delay variation and is often experienced by packets in IP networks.

**Chart 1: Delay Results for 2%, 10%, 20% dropping rates**

The results presented in Chart 1 are taken at three different constant dropping rates: 2%, 10%, and 20%. As the reliability requirement increases, delay increases as well. The graph shows the protocol's potential for high reliability requirements only when packet loss is low (here is between 0 - 10%). At high error rates video transmission can continue timely only when packet losses more than 5% can be tolerated. A reliable real-time service under the conditions presented by the first graph cannot be achieved.

Chart 2 below shows the throughput measurements of AOTP under various error conditions. Here, the protocol recovers completely from errors applying always its retransmission mechanism.

**Chart 2: AOTP behaviour under various dropping rates and full error recovery**

Packet video is jitter-intolerant traffic. Because of jitter, the receiver has to buffer the video frames until all major frames have arrived in order. The receiver can then deliver the frames to high level protocol to display the images. Any frames that arrive later are simply discarded because they are useless. The jitter buffer adds to overall delay. A shorter buffer means less delay, but there is a trade-off: set the buffer too short, and some of the frames can not arrive in time, which means dropped packets and clipped images; or, if we want to guarantee that all frames will be delivered, retransmission is a necessary recovery strategy. Consequently, further delay is added. In AOTP, the receiving rate and receiving percentage is continuously monitored: The jitter buffer size is dynamically adjusted. If the receiving rate and percentage are high, that means there is less jitter, the buffer size is reduced in order to reduce the delay. In case the receiving percentage is low, the jitter buffer is increased in order to guarantee image quality. However, buffer adjustments to current conditions do not favor a smooth playback. In AOTP, the buffer size can be adjusted to guarantee that the 'I' frame is received, that is, if 'I' frame is missing, retransmission is triggered, else missing frames can be ignored. On the other hand, the
buffer used to enhance smooth playback, can be of predetermined size. Then, the priority service of AOTP can be made to significantly increase the reliability level, and hence the application quality, by delivering the significant frames; thus, the application layer will need not to discard the already delivered B and P frames. It can be observed from Charts 3 & 4 that jitter is not increased significantly when a priority service is used (the buffer size was 150KB for these experiments). With reliability requirement 95% and dropping rate of 20%, the average delivery time of 1Mbyte was 18.75 seconds. For the same experiment without a priority control the average time was 18.008. The quality damage for the application in the latter case was significant: 5% data loss in addition to the discarded B/P frames; hence data loss can reach levels up to 15%. On the other hand, using a priority control service, the data loss was far less than 5% since all the missing I-frames have already been recovered. Finally, Chart 5 below shows that for this service, the impact of reliability requirement to the average delay is not significant.

5. Conclusion

Reliability/Throughput/Jitter tradeoffs can enhance the real-time capabilities of transport protocols. AOTP can be qualified as a transport protocol of choice for several multimedia applications. Using flexible error control strategies it can yield application-oriented quality of service achievements.

6. References


column | Test # | Protocol | Drop Rate | Requirement | Avg Time | Throughput |
--------|--------|----------|------------|------------|-----------|------------|
1       | 1      | AOTP     | 2%         | 100%       | 17.1      | 64220      |
2       | 2      | AOTP     | 10%        | 100%       | 22sec     | 47662      |
3       | 3      | AOTP     | 20%        | 100%       | 58.1sec   | 10047      |
4       | 4      | AOTP     | 20%        | 90%        | 14.3sec   | 73236      |
5       | 5      | AOTP     | 20%        | 80%        | 10.3sec   | 91800      |
6       | 6      | AOTP     | 5%         | 90%        | 12.8sec   | 81920      |
7       | 7      | AOTP     | 2%         | 90%        | 15.4sec   | 68089      |
8       | 8      | AOTPBaseline | 0%       | 100%       | 5sec      | 269715     |

Table 1: Throughput measurements under various reliability requirements.