Taxonomy and Survey of Retransmission Based Partially Reliable Transport Protocols

KARL-JOHAN GRINNEMO

Department of Computer Science
KARLSTAD UNIVERSITY
Karlstad, Sweden 2002

Taxonomy and Survey of Retransmission Based Partially Reliable Transport Protocols

KARL-JOHAN GRINNEMO

Technical Report no. 2002:34

Department of Computer Science
Karlstad University
SE–651 88 Karlstad, Sweden
Phone: +46 (0)54–700 1000

Contact information:
Karl-Johan Grinnemo
Signaling Product Planning & Support
System Management
Box 1038
Lagergrens gata 2
SE–651 15 Karlstad, Sweden

Phone: +46 (0)54–29 41 49
Email: karl-johan.grinnemo@tietoenator.com

Printed in Sweden
Karlstads Universitetstryckeri
Karlstad, Sweden 2002

Taxonomy and Survey of Retransmission Based Partially Reliable Transport Protocols

KARL-JOHAN GRINNEMO
Department of Computer Science, Karlstad University

Abstract

The mismatch between the services offered by the two standard transport protocols in the Internet, TCP and UDP, and the services required by distributed multimedia applications has led to the development of a large number of partially reliable transport protocols. That is, protocols which in terms of reliability places themselves between TCP and UDP. This paper presents a taxonomy for retransmission based, partially reliable transport protocols, i.e., the subclass of partially reliable transport protocols that performs error recovery through retransmissions. The taxonomy comprises two classification schemes: one that classifies retransmission based, partially reliable transport protocols with respect to the reliability service they offer and one that classifies them with respect to their error control scheme. The objective of our taxonomy is fourfold: to introduce a unified terminology; to provide a framework in which retransmission based, partially reliable transport protocols can be examined, compared, and contrasted; to make explicit the error control schemes used by these protocols; and, finally, to gain new insights into these protocols and thereby suggest avenues for future research. Based on our taxonomy, a survey was made of existing retransmission based, partially reliable transport protocols. The survey shows how protocols are categorized according to our taxonomy, and exemplifies the majority of reliability services and error control schemes detailed in our taxonomy.

Keywords: Taxonomy, survey, partial reliability, retransmission based, service, implementation, error control scheme

Contents

1 Introduction

The two standard transport protocols in the TCP/IP suite, TCP [29] and UDP [28], date back some thirty years. They were primarily designed to offer a service appropriate for the prevailing applications in those days. TCP was designed to offer a completely reliable service, a service suitable for such applications as e-mail, file transfer, and remote login; UDP, on the other hand, was designed to offer an unreliable service, a service appropriate for simple query-response applications such as name servers and network management systems. In recent years, we have witnessed a growing interest in distributed multimedia applications, applications typically requiring a service that in terms of reliability places itself somewhere between the services offered by TCP and UDP but that often have stringent demands on throughput, delay, and delay jitter. To address the needs of this class of applications, the Real-time Transport Protocol (RTP) [31] and other application support protocols have been proposed.

These protocols try to follow the principles of protocol architecture design outlined by Clark and Tennenhouse [9]. Specifically, they try to implement their two key architectural principles: application level framing and integrated layer processing. However, apart from giving the application more control over transport level decisions, e.g., error recovery decisions, these protocols also impose a number of transport level responsibilities on the application, e.g., flow and congestion control and the administration of the timing of the application messages.

The fact that RTP and similar protocols impose a number of duties on the application that are normally taken care of by the transport protocol and that are essentially of no interest to the application, have made many researchers argue that this is not the right way to go [21]. They re-emphasize the requirement that there should be a clean separation between the *specification* and *implementation* of a service. In particular, they argue for a transport protocol which, in the same way as RTP, gives the application more influence on the transport level decisions, but which, in contrast to RTP, relieves the application from all transport level responsibilities. To this end, a class of transport protocols has emerged that offer a service enabling the application to trade some reliability for improved performance with respect to some or all of the metrics: throughput, delay, and delay jitter. In other words, emerged to offer a service more suitable for multimedia and other real-time applications than either TCP or UDP. This class of protocols is commonly called *partially reliable transport protocols*.

From an implementation perspective, we may differentiate between two major classes of partially reliable transport protocols: *closed-loop* and *open-loop* protocols. Closed-loop protocols dynamically adapt their error control scheme based on feedback of the current network conditions. In contrast, open-loop protocols do not use any feedback from the network. Instead, they work by adding redundancy to the payload and thereby mitigate the impact of losses. While open-loop protocols have been extensively studied and used for multimedia communication in the last twenty years [5, 6], closed-loop techniques have not been considered appropriate for multimedia communication

for more than half that time.

At the beginning of the 1990s, Dempsey [12], Papadopoulos and Parulkar [26], and others demonstrated the feasibility of using retransmission-based closed-loop partially reliable transport protocols for multimedia communication. Since then, a large number of retransmission based, partially reliable transport protocols for time sensitive applications such as multimedia has been proposed.

The main purpose of this paper is to present a taxonomy for retransmission based, partially reliable transport protocols. The taxonomy comprises two classification schemes: one that classifies retransmission based, partially reliable transport protocols with respect to the reliability service they offer and one that classifies them with respect to their error control scheme.

The taxonomy has been developed through an extensive analysis of a significant number of existing retransmission based, partially reliable transport protocols. We believe that it not only gives valuable insights into the key components of the existing protocols but that it also provides a unified terminology and a baseline for future protocols.

Our taxonomy has been inspired by earlier work of, among others, Diaz and Dempsey. In particular, our view of the concept of a partially reliable service is to a large extent derived from the work on partially reliable and partially ordered services done by Diaz et al. [14]; our classification of retransmission based, partially reliable transport protocols with respect to their error control scheme is to some degree influenced by the discussion of closed-loop, partially reliable transport protocols in Dempsey's thesis [11].

This paper also provides a classification and survey of existing protocols. The paper shows how a selection of existing protocols are classified with respect to our taxonomy. Furthermore, the paper presents a survey of a subset of the classified protocols. The subset of protocols covers the majority of reliability services and error control schemes in our taxonomy.

This paper is organized as follows. Section 2 provides some preparatory material. In particular, the concept of a retransmission based, partially reliable transport protocol is defined. Our taxonomy is presented in Section 3. Section 4 gives a survey of existing retransmission based, partially reliable transport protocols and shows how they are classified with respect to our taxonomy. Finally, Section 5 concludes the paper with a brief summary of our proposed taxonomy and a discussion of the insights gained in developing the taxonomy. The section also briefly considers how these insights can be used in the design of future retransmission based, partially reliable transport protocols.

2 Preliminaries

The taxonomy of retransmission based, partially reliable transport protocols presented in this paper is based on the following definition of a *partially reliable service*.

Definition 1 *Let r denote the reliability level offered by a service. A service is consid-*

ered partially reliable provided $r \in]0\%, 100\%[$.

On the basis of Definition 1, we used the following definition of a *retransmission based, partially reliable transport protocol*.

Definition 2 *A retransmission based, partially reliable transport protocol is a transport protocol that is explicitly designed to offer a partially reliable service and that uses an error control scheme in which error recovery is performed through retransmissions.*

The reason Definition 2 specifically states that a transport protocol must be explicitly designed to offer a partially reliable service in order to be a partially reliable transport protocol is that, from a practical viewpoint, there is no such thing as a transport protocol offering a completely reliable service. For example, TCP is designed to offer a completely reliable service but is only able to do so as long as some conditions hold. In particular, TCP depends on the ability of the IP protocol to provide end-to-end connectivity and assumes that the TCP checksum is able to detect all conceivable kinds of bit errors. Consequently, TCP fails to provide a completely reliable service in the rare occasions when the network becomes partitioned or the TCP checksum fails to detect a corrupted packet [35].

3 The Taxonomy

As mentioned in Section 1, our taxonomy consists of two classification schemes: one which classifies retransmission based, partially reliable transport protocols with respect to their offered *reliability service* and one which classifies them with respect to their *error control scheme*. The former classification scheme is presented in Subsection 3.1 and the latter in Subsection 3.2.

3.1 Classification with Respect to Reliability Service

Figure 1 depicts the classification scheme with respect to the reliability service offered. As follows from Figure 1, the protocols are classified along three dimensions: *specification of reliability level*, *granularity*, and *adaptiveness*.

Specification of Reliability Level. We distinguish between two main classes of specifications: *implicit* specifications and *explicit* specifications. In implicit specifications, the reliability level is implicitly determined by one or more QoS and/or transport constraints other than just the guaranteed reliability level, e.g., number of retransmissions, priorities, deadlines, playback rate, or maximum latency. Contrary to implicit specifications, explicit specifications explicitly prescribe a certain guaranteed reliability level. Common ways of formulating explicit reliability level requirements include: specifying an upper bound for packet loss, stating the minimum number of messages that must be successfully transmitted out of a fixed-length message group, or assigning reliability classes to messages.

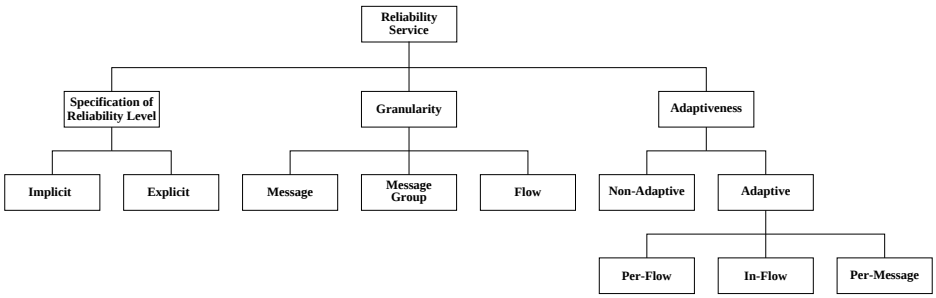

Figure 1: Classification with respect to reliability service.

Granularity. The dimension granularity refers to the calculation base of the reliability service provided by a retransmission based, partially reliable transport protocol. Or, put differently, the granularity of a retransmission based, partially reliable transport protocol represents the smallest unit of data on which the protocol controls the reliability service, and consequently is the smallest unit of data on which the protocol ascertains the reliability service. In implementation terms, the dimension granularity refers to the unit of data considered by the error control scheme when it makes retransmission decisions on lost or corrupted packets (cf. Section 3.2). As shown in Figure 1, we distinguish between three main classes of protocols with respect to granularity: *message*, *message group*, and *flow*.

Message. In this context, the term message denotes messages exchanged between application entities, i.e., Application Protocol Data Units (APDUs). When a protocol has a granularity of a message, this means that the error control scheme of the protocol performs retransmissions based on the status of one message. This, for example, can be that the reliability service is based on the number of times a message has been retransmitted, or that the reliability service is based on a priority level assigned to a message.

Typically, protocols with a granularity of a message target video and audio streaming applications: both video and audio coders usually generate sequences of fixed-sized data blocks that conveniently fit into messages. Furthermore, video coding schemes such as MPEG-1, MPEG-2, and H.263 generate data blocks of different importance, e.g., H.263 generates I-, P-, and B-frames of which I-frames are more important than either the P- or B-frames. By using a protocol with a granularity of a message, a video application using one of these video coding schemes can exploit the fact that all data blocks are not equally important and assign different reliability levels to each data block.

Message Group. A protocol is said to have a granularity of a message group when

the retransmission decision of the protocol involves not only one message but a pre-specified number of messages. Typically, a protocol adhering to this class, calculates the reliability level by counting the number of successfully received messages within a fixed-length message group.

Protocols with a granularity of a message group primarily target the same niche of applications as those with a granularity of a message, i.e., video and audio streaming applications. Compared to protocols with a granularity of a message, the major advantages of message-group based protocols are that they are generally easier to realize and, owing to their coarser granularity, entail less overhead. However, the coarser granularity of message group based protocols is not always beneficial. While, it could be argued that a granularity of a message group is sufficient for many audio streaming applications, it is less than ideal for video streaming applications: while audio coders typically produce packets of more or less the same importance, this is, as said, generally not the case with video coders.

Flow. A flow refers to a unidirectional stream of messages from a single media source. One or several flows constitute a session: a single connection and/or conversation between one endpoint and one or several other endpoints. Common examples of flows are video streams in a video broadcast application and audio streams in an Internet radio application. When a protocol belonging to this class of protocols makes a retransmission decision at a particular time during a flow, it bases its retransmission decision on all packets sent and/or received in this flow up to this time.

To our knowledge, the number of protocols having a granularity of a flow are very scarce. As a matter of fact, we know of only two protocols: SRP [27] and PRTP-ECN [17, 18]. SRP primarily targets audio applications, and PRTP-ECN has been used to improve the transmission of images on Web pages. The principal incentive for these two protocols to use a granularity of a flow is that it makes them very easy to implement: Compared to message- and message-group-based protocols, flow-based protocols tend to involve less state information, and are because of that often less complex. However, flow based protocols tacitly assume that all messages are of equal importance. Therefore, special arrangements have to be made in order to use these protocols for transmission of images and video streams. For example, PRTP-ECN only works for images that are coded according to some robust image coding scheme.

Adaptiveness. The last dimension, adaptiveness, refers to the situation in which, during the lifetime of a session, a protocol permits an application to change its reliability requirements. There are two main classes of protocols with respect to adaptiveness: *non-adaptive* and *adaptive*.

Non-Adaptive. Those protocols, which offer a pre-configured reliability service,

are members of the non-adaptive class of protocols. In other words, non-adaptive protocols comprise protocols which, in the same way as TCP and UDP, are designed to offer a certain fixed reliability service. However, in contrast to TCP and UDP, they are rarely general protocols. They are often specifically tuned for a certain category of applications, or even a certain product.

Adaptive. The adaptive class is the complement of the non-adaptive class, i.e., comprises all protocols that do not offer a pre-configured reliability service. It has three subclasses: *per-flow* adaptive, *in-flow* adaptive, and *per-message* adaptive. In per-flow adaptive protocols, the reliability service is specified at the inception of a flow and remains fixed during the whole lifetime of the flow. A further degree of flexibility is provided by in-flow adaptive protocols, which enables applications to alter the reliability service during the flow lifetime. Finally, the highest degree of flexibility is found in in-flow adaptive flows that permit applications to alter the reliability service on a per-message basis.

Since per-flow adaptive protocols make no provisions for an application to re-negotiate the reliability service of a flow, they are primarily useful for inelastic applications such as audio broadcasting. More elastic applications, such as many video streaming tools, are typically better off with an in-flow or per-message adaptive transport protocol: at high traffic loads, an in-flow or per-message adaptive protocol permits an application to continue a flow, although at a lower service level.

As follows from the definition of in-flow adaptive flows, they represent a middleway between per-flow adaptive and per-message adaptive protocols. There are very few in-flow adaptive protocols. However, an example of an in-flow adaptive protocol is PRTP-ECN [17, 18].

3.2 Classification with Respect to Error Control Scheme

This section explains how, according to our taxonomy, retransmission based, partially reliable transport protocols are classified with respect to their error control scheme: Subsection 3.2.1 discusses the logical architecture of an error control scheme of a retransmission based, partially reliable transport protocol. On the basis of the logical architecture, the actual classification scheme is presented in Subsection 3.2.2.

3.2.1 The Logical Architecture of an Error Control Scheme

The error control scheme of a retransmission based, partially reliable transport protocol can be viewed as consisting of five principal components: an *error detection component*, an *error feedback component*, a *retransmission-decision component*, a *retransmission-decision feedback component*, and finally, a *retransmission component*. Figure 2 illus-

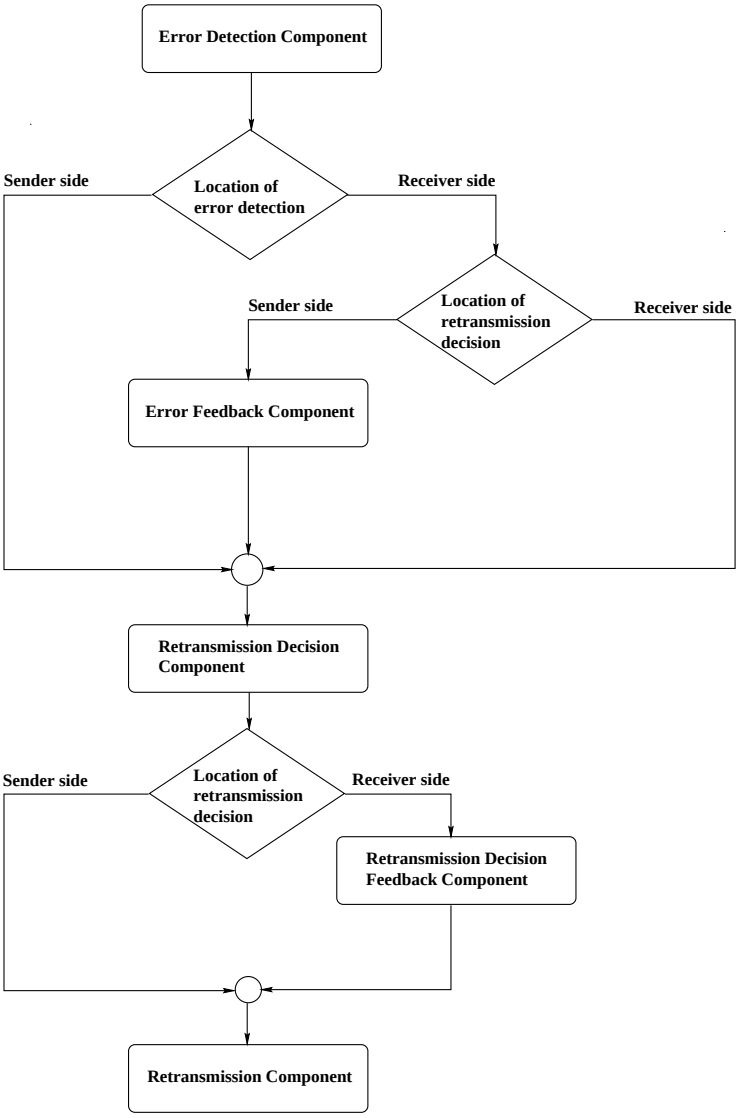

Figure 2: The logical architecture of an error control scheme.

trates the interrelationship between the five components, and the responsibility of the respective component is as follows:

Error Detection Component. The error detection component is responsible for detecting lost or corrupted packets. Packet losses are usually detected through the reception of out-of-sequence packets and/or timeouts, while packet corruption is discovered through the use of checksums.

Error Feedback Component. In error control schemes in which the error detection and the retransmission decision are not made on the same side, i.e., error detection occurs at the receiver side while the retransmission decision takes place at the sender side, an error feedback component is used to inform the retransmission decision component about packet loss and corruption events.

Retransmission Decision Component. As follows from its name, the retransmission decision component comprises the part of the error control scheme that is responsible for deciding whether a lost or corrupted packet should be retransmitted.

Retransmission Decision Feedback Component. The retransmission decision feedback component is only needed in error control schemes in which the retransmission decision takes place at the receiver. It is the responsibility of this component to communicate the retransmission decisions of the receiver back to the retransmission component at the sender.

Retransmission Component. The retransmission component is always located at the sender. It is the responsibility of this component to execute the retransmission requests made by the retransmission decision component, i.e., it is this component which performs the actual retransmissions.

3.2.2 The Classification Scheme

Retransmission based, partially reliable transport protocols differ from completely reliable ones primarily in their retransmission strategy. Thus, as shown in Figure 3, the classification scheme with respect to the error control scheme is exclusively based on the most salient features of the retransmission decision component. In particular, classification is done along the two dimensions of *location* and *decision base*.

Location. The retransmission decision component of an error control scheme is located at either the sender side or the receiver side. Depending on the location of the retransmission decision component, we distinguish between *sender based* protocols and *receiver based* protocols.

Although an analytical study by Marasli et al. [23] suggests that sender based protocols in some instances exhibit a slightly better throughput performance than receiver based protocols, the latter indeed possess some attractive features. Receiver based protocols are generally more scalable and versatile. An example is the case in which a server serves several clients, all with different reliability service requirements. In the sender based case, the responsibility of providing the

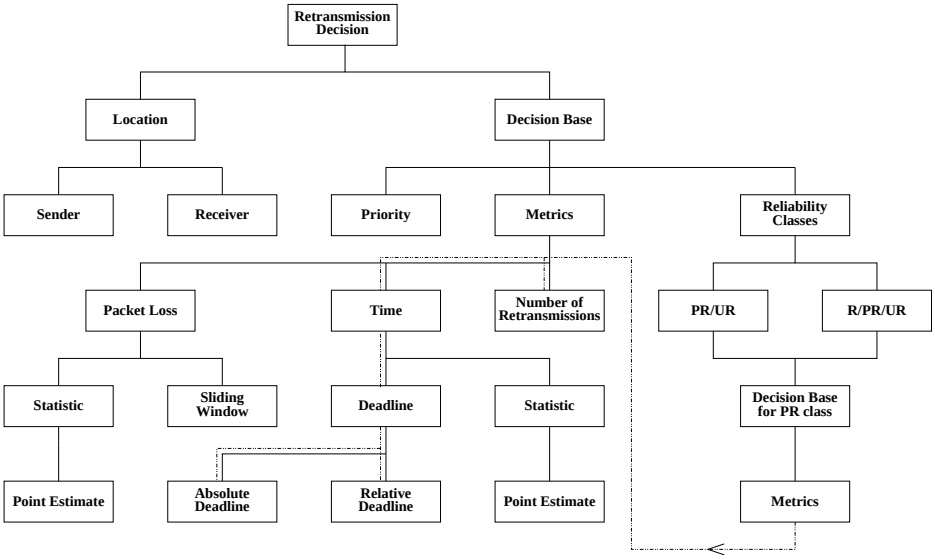

Figure 3: Classification with respect to error control scheme.

correct service to all clients is primarily the server's. In the receiver based case, the responsibility has to a large degree been distributed to each one of the clients. Furthermore, in the receiver based case, the sender is not involved in the retransmission decisions and is consequently able to simultaneously serve several clients with different retransmission decision components.

In some protocols, the retransmission decision is normally made by the receiver but in exceptional cases is ignored by the sender. For example, in the CM protocol proposed by Papadopoulos and Parulkar [25], the sender has a retransmission buffer whose size approximately follows the size of the playout buffer at the receiver side. When a packet has been sent, it is placed in the retransmission buffer and, if necessary, the oldest packet in the retransmission buffer is discarded. When a retransmission request arrives at the sender for a packet that has already been discarded from the retransmission buffer, the retransmission request is ignored by the sender. Even though it could be argued that protocols like the CM protocol should be classified as hybrid sender based receiver based protocols, we classify them as receiver based protocols because only the retransmission decision at the receiver side is made with regard to the requirements imposed by the data stream, i.e., uses a decision base that is either metrics based, priority based, or based on reliability classes.

Decision Base The decision base comprises the metrics, rules, and/or heuristics that

form the basis for the retransmission decisions made by a retransmission decision component. This dimension has three main classes: *priority*, *metrics*, and *reliability classes*.

***Priority*.** In priority based protocols, each packet is assigned a priority based on its relative importance. Retransmission of lost packets is always performed in priority order with high priority packets sent before packets with lower priorities. Often protocols are not pure priority based protocols but are combined priority and deadline based. Apart from having a priority, packets in combined priority- and deadline-based protocols also have a deadline to meet. In these protocols, lost packets are typically retransmitted in priority order until they have been successfully received or their deadline has expired. This means that high priority packets have a better chance of being delivered since, in the event of packet loss, they are more likely to be retransmitted.

Priority based protocols are very lightweight and have therefore found their use in the area of audio and video broadcasting. For example, the CUDP [32] transport protocol targets audio and video file servers and SR-RTP [15] targets video file servers.

***Metrics*.** Protocols that base their retransmission decision on direct or indirect measurements of one or several properties of a flow, e.g., packet loss rate or timeliness, are considered metrics based protocols. We distinguish between three classes of metrics based protocols: protocols that base their retransmission on some kind of estimate of the *packet loss rate*; protocols that perform retransmissions as long as the *timeliness* of the data flow is not violated; and protocols that indirectly consider the timeliness of the data flow by restricting the *number of retransmissions* performed on individual messages.

Packet Loss. To our knowledge, existing protocols use either of two approaches to estimate the packet loss rate. The first approach involves using a *statistic*, e.g., the arithmetic average or mean packet loss rate. Protocols belonging to this class usually monitor the packet loss rate by continuously calculating an estimate or weighted estimate of the mean packet loss rate. The second approach involves calculating the average packet loss rate over a fixed-length sequence of packets. Since the packet loss based protocols using this approach typically keep track of the fixed-length sequence of packets through the use of a sliding window mechanism, this subclass of packet-loss-based protocols is called the *sliding window* class.

A problem with pure packet loss based protocols are their obliviousness to time, which makes them not altogether perfect for audio, video, and other time sensitive applications. However, measurements made on a pure packet loss based protocol, AOEC [16], suggest that the improve-

ments obtained in terms of throughput and delay by selective retransmissions compensates for this problem to some degree.

Time. Time based protocols comprise those protocols that employ a metric that is a function of time. The class of time based protocols can be further divided into the subclasses: *deadline based* and *statistic based* protocols.

In deadline based protocols, the retransmission decision component issues a retransmission request for a lost packet provided the retransmitted packet is likely to meet the deadline of the lost packet. Typically, deadline based protocols are used by streaming media applications where it is important that packets arrive at the receiver before their playout time. There are two major classes of deadline based protocols: *absolute deadline based* and *relative deadline based* protocols. Absolute deadline based protocols refer to protocols in which deadlines are calculated with respect to the beginning of a flow while, in relative deadline based protocols, the deadline of a packet is calculated relative to preceding packets.

The class of statistic based protocols comprises all time based protocols that use a metric that is a function of statistical estimates of one or several time related performance metrics, e.g., mean latency.

As far as we know, there exist no purely time-statistic based protocols. However, there is at least one protocol, SRP [27], that uses a metric involving both packet loss and time. In contrast to deadline based protocols, SRP is able to make explicit trade-offs between reliability level and timeliness.

Number of Retransmissions. Contrary to time based protocols, protocols belonging to this class have no notion of time. Instead, they impose timely delivery of packets by limiting the number of times a packet can be retransmitted. More specifically, a retransmission counter is associated with each packet. The counter is decreased every time the packet is retransmitted. When the retransmission counter of a packet reaches zero, the packet is discarded.

There are very few protocols that adhere to this class. In fact, we have not found a single protocol that has a decision base only involving the number of times a packet has been retransmitted. On the other hand, there is a reliability class based protocol, k-XP [3, 22], in which one reliability class has a decision base consisting of the number of times a packet has been retransmitted.

The reason there are so few protocols that base their retransmission decision on the number of times a packet has been retransmitted is of course that this is a very imprecise metric; it only indirectly governs the packet loss rate and the timeliness. However, in combination with reli-

ability classes, it has proven itself quite useful. In particular, using the number of times a packet has been retransmitted as a decision base has been proven useful in implementing a better than best effort reliability class similar to the controlled load service class of IntServ [7].

Reliability Classes. In reliability class based protocols, each packet is associated with one of a pre-defined set of reliability classes. The decision of whether a lost or corrupted packet should be retransmitted is solely based on the class of the packet. For example, packets associated with a reliable service class are retransmitted until they have been successfully delivered, while packets that are members of an unreliable service class are never retransmitted.

The majority of reliability class based protocols use either of two reliability class schemes: *PR/UR* or *R/PR/UR*. The PR/UR scheme comprises two service classes: a partially reliable (PR) service class and an unreliable (UR) service class, while the R/PR/UR scheme also offers a reliable (R) service class.

From a logical point of view, the PR service class of a reliability class scheme has its own decision base and therefore theoretically the same subclasses as the decision base dimension of the retransmission-decision component. However, in practice, the decision base of the PR service classes in existing protocols are primarily metrics based. The decision base here is normally deadline based or takes the form of an upper limit on the number of times a packet can be retransmitted.

A major advantage with reliability class based protocols is that they provide for explicit synchronization between heterogeneous flows, i.e., flows with different reliability and timing requirements. Thus, video conferencing and similar applications that involve several heterogeneous flows are relieved from the complex task of performing synchronization between multiple independent flows.

4 A Classification and Survey of Existing Protocols

This section surveys a selection of existing retransmission based, partially reliable transport protocols and classifies them with respect to our taxonomy (see Section 3). Table 1 shows how the following protocols are classified: Slack ARQ [13], PECC [11, 12], k-XP [3, 22], POCv2 [10], AOEC [16], CUDP [32], VDP [8], Jacobs/Eleftheriadis [20], TLTCP [24], SRP [27], Papadopoulos/Parulkar [25], SR-RTP [15], HPF [21], MSP [19], XUDP [4], PR-SCTP [33], and PRTP-ECN [17, 18]. A survey of a subset of these protocols is also presented: PECC, POCv2, SRP, HPF, PR-SCTP, and PRTP-ECN. Together, this subset illustrates the majority of reliability services and error control techniques detailed in our taxonomy.

	Reliability Service			Retransmission Decision	
Protocol	Specification of Reliability Level	Granularity	Adaptiveness	Decision Base	Location
Slack ARQ	Implicit	Message	Per-Flow	Absolute Deadline	Receiver
PECC	Explicit	Message Group	Per-Flow	Absolute Deadline, Sliding Window	Receiver
k-XP	Explicit	Message	Message	Reliability Classes, R/PR/UR(Number of Retransmissions)	Sender
POCv2	Explicit	Message	Message	Reliability Classes, R/PR/UR(Relative Deadline)	Receiver
AOEC	Explicit	Message Group	Per-Flow	Sliding Window	Receiver
CUDP	Implicit	Message Group	Per-Flow	Priority	Sender
VDP	Implicit	Message	Per-Flow	Reliability Classes, PR/UR(Absolute Deadline)	Receiver
Jacobs/Eleftheriadis	Implicit	Message	Per-Flow	Absolute Deadline	Receiver
TLTCP	Implicit	Message	Message	Absolute Deadline	Sender
SRP	Explicit	Flow	Per-Flow	Point Estimate of Packet Loss and Time	Receiver
Papadopoulos/Parulkar	Implicit	Message	Per-Flow	Absolute Deadline	Receiver
SR-RTP	Implicit	Message	Per-Flow	Absolute Deadline, Priority	Receiver
HPF	Explicit	Message	Per-Flow	Reliability Classes, R/PR/UR(Absolute Deadline)	Sender
MSP	Implicit	Message	Per-Flow	Absolute Deadline	Receiver
XUDP	Explicit	Message	Message	Reliability Classes, R/PR/UR(Absolute Deadline)	Sender
PR-SCTP	Implicit	Message	Message	Absolute Deadline	Sender
PRTP-ECN	Explicit	Flow	In-Flow	Point Estimate of Packet Loss	Receiver

Table 1: Classification of retransmission based, partially reliable transport protocols in our taxonomy. The decision bases of the partially reliable classes in the reliability class schemes are appended in parentheses to the names of the schemes.

4.1 PECC

To be precise, PECC (Partially Error-Controlled Connection) is an error control scheme, not a protocol. Developed by Dempsey et al. [11, 12] as an extension to the multi-service transport protocol Xpress Transfer Protocol [36] (XTP), it was primarily intended for continuous media applications.

The application atop PECC specifies its service requirements explicitly on a per-flow basis through four parameters: `fifo_min`, `window_length`, `window_density`, and `max_gap`. The `fifo_min` parameter indicates the minimum number of contiguous bytes that must be queued before the receiver is permitted to issue a retransmission request. In other words, `fifo_min` should be an estimate of the number of bytes consumed by the application during one round trip time. The two parameters of `window_length` and `window_density` specify the loss tolerance of the application. They specify that no more than `window_density` bytes are permitted to be lost out of `window_length` bytes of application data. From a practical viewpoint, this means that a reliability service at a granularity of a message group is offered: setting `window_length` to a number of bytes equal to or less than the size of a message is not meaningful[1]. Finally, the last parameter, `max_gap`, puts a limit on burst losses, giving an upper bound to how many contiguous bytes are permitted to be lost by the application.

Since PECC is mainly intended for continuous media applications, it assumes that the XTP receiver logically places received data in a FIFO buffer that is emptied at an approximately constant rate (isochronously). Consequently, the `fifo_min` parameter serves as an absolute deadline for the data received. Together with the parameters `window_length` and `window_density`, this makes PECC an absolute deadline and sliding-window based protocol.

In PECC, the retransmission decision takes place at the receiver side. Every time an out-of-sequence packet is received, this is taken as an indication of one or more packets being lost and results in the invoking of the retransmission decision component of PECC. If there are more than `fifo_min` bytes in the FIFO buffer, a retransmission request is issued for the presumably lost data. Otherwise, PECC tries to skip as much data as is needed to facilitate a retransmission, i.e., make the depth of the FIFO buffer greater than `fifo_min` bytes. However, when this is not possible without violating the `max_gap` parameter or the sliding window determined by the `window_length` and `window_density` parameters, PECC reports a failure to the application and skips the data anyway.

4.2 POCv2

POCv2 was proposed by Conrad et al. [10] at the University of Delaware in an attempt to design a transport protocol better suited for distributed multimedia applications. How-

[1] Notably, if `window_length` is assigned a value equal to or less than the size of a message, an unreliable service is obtained when `window_density` is set to 0, and a completely reliable service is obtained for all values of `window_density` greater than 0.

ever, the origin of POCv2 traces back to the POC [1, 2] (Partial Order Connection) protocol, which was developed as a joint effort between the University of Delaware and LAAS/ENSICA.

Apart from being a partially reliable transport protocol, POCv2 builds upon the notion of partial order. It considers a flow as consisting of a partially ordered sequence of messages, where each message corresponds to exactly one media object (e.g., an audio clip or a component of a video frame).

POC is a reliability class based protocol: The POCv2 application decomposes a flow into messages; specifies the partial order of the messages; and assigns each message to one of three reliability classes: reliable, partially reliable, or unreliable. Furthermore, during a flow, POCv2 enables an application to alter the reliability class assigned a particular message.

The retransmission decisions in POCv2 are taken by the receiver. Whether or not a request for retransmission of a message should be issued depends on the reliability class of the message: For reliable messages, retransmission requests are issued until they are successfully received; partially reliable messages are retransmitted as long as the receiver has some messages to deliver to the application. The receiver will stop issuing retransmission requests for a partially reliable message when there are no more messages to deliver to the application, and when declaring the partially reliable message as lost will make some messages succeeding the lost message (with respect to the partial order among the messages) deliverable. In other words, the decision base of the partially reliable class of messages is relative deadline based, where the deadlines for the partially reliable messages are measured with respect to their preceding messages in the partial order. Finally, no retransmission requests are issued for unreliable messages.

4.3 SRP

Commonly, retransmission based partially reliable transport protocols that consider both the timing and the packet loss requirements of a multimedia streaming application, e.g., PECC, do so by simply giving the timing requirements priority over the packet loss requirements; lost packets are retransmitted as long as this can be done without violating the timing requirements. In contrast, the SRP (Selective Retransmission Protocol) protocol proposed by Piecuch et al. [27] not only strives to offer a service that complies with the timing and packet loss requirements imposed by the streaming application but also strives to offer the service that gives the optimal trade-off between the two requirements.

During an SRP session, the SRP client is responsible for receiving a multimedia stream from the server and issuing retransmission requests if necessary. The retransmission decisions of the receiver are governed by the maximum tolerable transmission delay and the maximum tolerable packet loss rate of the application. These performance requirements are communicated to SRP by the application at the inception of a flow and are specified on a per-flow basis.

The SRP receiver assumes that packets arrive at a constant rate. The receiver thus

maintains an estimate of the arrival time of the next packet in a flow. A packet is considered lost if it has not been received before its expected arrival time has elapsed.

The expected arrival time of a packet is calculated as the arrival time of the previous packet plus the round trip timeout (RTO) value maintained by the SRP receiver. Since it is vital for the performance of SRP that the RTO is accurate, the receiver updates the RTO by sending time probe packets to the sender at regular intervals.

The SRP receiver implements two retransmission decision algorithms: Equal Loss Latency (ELL) and Optimum Quality (OQ). ELL and OQ are based on the notions of loss ratio and delay ratio. The loss ratio, r_{loss}, is the quotient of the current packet loss rate divided by the maximum tolerable packet loss rate, and the delay ratio, r_{delay}, is the quotient of the current transmission delay divided by the maximum tolerable transmission delay. As an estimate of the current transmission delay, one-half of RTO is used. When a packet loss is detected, ELL decides whether the lost packet should be retransmitted based on which case minimizes Equation 1, while OQ tries to minimize Equation 2.

$$\Delta = |r_{loss} - r_{delay}| \quad (1)$$

$$\Delta = r_{loss}^2 + r_{delay}^2 \quad (2)$$

4.4 HPF

The HPF (Heterogeneous Packet Flows) protocol was designed by Li et al. [21] to effectively support heterogeneous packet flows: for example, MPEG flows with frames of different priority or multiplexed audio/video streams. The primary motivation for HPF was to demonstrate the feasibility of designing a transport protocol that provides mechanisms for flow and congestion control on a per-flow basis and mechanisms for reliability, sequencing, framing, and prioritization on a per-message basis.

The application atop HPF partitions the data stream into messages, e.g., MPEG frames, and specifies the service requirements at the initiation of a flow on a per-message basis. In particular, HPF enables an application to assign a message to one of the three reliability classes: reliable, unreliable, and unreliable delay bounded. The application messages are then treated as single entities by HPF, i.e., all packets belonging to the same application message are assigned the same reliability class as the message.

HPF is a sender based protocol, i.e., the retransmission decisions are made by the sender. The retransmission policy for reliable and unreliable packets is simple: a reliable packet is retransmitted until it has been successfully received, while an unreliable packet is never retransmitted. The retransmission policy for unreliable delay bounded packets are somewhat more complex. Specifically, all delay bounded packets are assigned a deadline based on the transmission rate. A packet is only retransmitted if the estimated round trip time suggests that the packet can be retransmitted and still meet its deadline. In other words, delay bounded packets use an absolute deadline based decision base.

4.5 PR-SCTP

To address the shortcomings and limitations of TCP and UDP for transportation of telephony signaling messages, the SIGTRAN (Signaling Transport) working group in IETF developed SCTP [34].

SCTP provides a message based, reliable, and ordered transport service to an application. It accomplishes this by fragmenting the application messages into so called chunks and assigning a Transmission Sequence Number (TSN), to each chunk. In the same way as in TCP, SCTP employs a cumulative acknowledgement mechanism: data chunks received are acknowledged by informing the sender of the TSN of the next expected chunk.

While the development of SCTP was directly motivated by the transfer of the SS7 (Signaling System Number 7) signaling protocol to IP, SIGTRAN ensured that the design of SCTP was general enough for the protocol to be suitable for applications with similar requirements. As a result of this design decision, a number of extensions to SCTP have been proposed [33, 37]. One of the most recent ones is PR-SCTP [33].

PR-SCTP is SCTP extended with a framework for implementing partially reliable transport services. It entails adding two new items to SCTP: a new parameter and a new type of chunk. The parameter introduced by PR-SCTP is used during the initialization of an SCTP session by both sides to signal support for PR-SCTP to the other side. A PR-SCTP session can only be initiated if both sides use PR-SCTP. If either the sending or receiving side is not using PR-SCTP, the other side has the option to end the session or start an SCTP session instead. The new type of chunk introduced by PR-SCTP, Forward Cumulative TSN (FCTSN), is used by the sender to inform the receiver that it should consider all chunks having a TSN less than a certain value as having been received.

At present, only one service has been proposed that is based on PR-SCTP, timed reliability. When an application uses this service, it assigns deadlines to its messages. In contrast to HPF, the deadlines are assigned continuously during the lifetime of a flow, and not only at the flow inception. These deadlines are translated into chunk lifetimes by PR-SCTP. Before a packet is transmitted or retransmitted, the PR-SCTP sender evaluates the lifetime of the packet. When the lifetime of a packet has expired, it is discarded, and the sender informs the receiver about this by sending it an FCTSN.

4.6 PRTP-ECN

PRTP-ECN is an extension to TCP suggested by Grinnemo et al. [17, 18] that aims to make TCP better suited for applications with soft real-time constraints (e.g., best effort multimedia applications). In particular, PRTP-ECN is an attempt to make this traditionally congestion insensitive class of applications aware of congestion. An attractive feature of PRTP-ECN is that it only entails modifying the retransmission decision component of TCP on the receiver side; the sender side is left unaffected.

PRTP-ECN offers a flow based reliability service. As long as no packets are lost in a flow, PRTP-ECN behaves in the same way as standard TCP. When an out-of-sequence

packet is received, however, this is taken as an indication of packet loss, and the modified retransmission decision component is invoked. This component decides, on the basis of the success rate of all previous packets in a flow, whether to acknowledge all packets up to and including the out-of-sequence packet or to do the same as standard TCP, i.e., acknowledge the last successfully received in-sequence packet and wait for a retransmission.

The success rate of previous packets, called the *current reliability level* (crl), is calculated as an exponentially weighted moving average over all packets up to but not including the out-of-sequence packet. It is calculated as

$$crl(n) = \frac{\sum_{k=1}^{n} af^{n-k} * p_k * b_k}{\sum_{k=1}^{n} af^{n-k} * b_k}\ , \tag{3}$$

where n is the sequence number of the packet preceding the out-of-sequence packet, af is the weight or *aging factor*, and b_k denotes the number of bytes contained in packet k. Variable p_k is given a binary value. If the kth packet was successfully received, then $p_k = 1$ otherwise $p_k = 0$.

An application communicates its lower bound reliability level through the aging factor and a second parameter called the *required reliability level* (rrl). This parameter functions as a target value. As long as $crl(n) \geq rrl$, lost packets are acknowledged. However, if an out-of-sequence package is received at a time when $crl(n)$ is below rrl, the last in-sequence packet is acknowledged, forcing the sender to retransmit the lost packet. It should be noted that an application is permitted to alter either af or rrl at any time during the lifetime of a flow, i.e., PRTP-ECN is an example of an in-flow adaptive protocol.

Although, PRTP-ECN is a flow based protocol, it has some features in common with message group based protocols. In particular, PRTP-ECN does not consider all messages as being equally important, which is usually the case with flow based protocols. Instead, the aging factor makes PRTP-ECN consider newly arrived messages to be more important than messages that arrived for some time ago. Furthermore, to some degree, PRTP-ECN provides for an application to control the maximum tolerable message burst loss length: More or less the same reliability level is given by several different combinations of af and rrl. However, the combinations differ from each other in that they translate to different upper limits on message burst loss.

In order to decouple the error control and congestion control schemes of TCP, PRTP-ECN uses the transport-level flags of ECN [30] (Explicit Congestion Notification). In particular, when lost packets are acknowledged, PRTP-ECN signals congestion to the sender side by setting the explicit congestion notification flag in the acknowledgement packet. As a result, when the sender receives an acknowledgement of a lost packet, it will act as if a congestion has occurred, e.g., reduce its congestion window; however, it will not re-send the lost packet.

5 Concluding Remarks

This paper presents a taxonomy for retransmission based, partially reliable transport protocols. A selection of existing protocols was also surveyed and classified according to the taxonomy.

The proposed taxonomy comprises two classification schemes. The first classification scheme classifies retransmission based, partially reliable transport protocols with respect to the reliability service they offer. In this scheme, retransmission based, partially reliable transport protocols are classified along three dimensions: specification of reliability level, granularity, and adaptiveness. The second classification scheme classifies retransmission based, partially reliable transport protocols with respect to their retransmission decision component. The scheme comprises two dimensions: location and decision base.

In the introduction to this paper, a clean separation between the service interface and implementation was used as an argument for the introduction of partially reliable transport protocols. However, the taxonomy and survey suggest that, in a large number of retransmission based, partially reliable transport protocols, the service interface is more or less transparent, i.e., closely coupled to the implementation of the error control scheme, and that very often the service interface and implementation are intertwined. In view of this, future protocols should probably consider using service interfaces that enable applications to specify their requirements in terms of meaningful performance metrics, not in terms of parameters of their error control scheme. For example, a video streaming application should be able to specify the playout rate and not be forced to specify the deadline of each individual message.

The taxonomy shows that the majority of retransmission based, partially reliable transport protocols use error control schemes that are simply variations of a relatively small set of core principles. Most protocols base their retransmission decision on one of the following core principles or a combination thereof: an estimate of the average packet loss rate, an absolute deadline, an upper bound on the number of retransmissions, priorities, or reliability classes. Noteworthy, neither of the core principles for the error control schemes of retransmission based, partially reliable transport protocols explicitly involve delay jitter, a performance parameter that for many multimedia applications is at least as important as the delay itself. Instead, delay jitter is almost always indirectly controlled through buffers at the receiver side. Based on this observation, we think that future retransmission based, partially reliable transport protocols may gain from, not only considering packet loss rate and delay, but also more actively try to control the delay jitter.

References

[1] P. D. Amer, C. Chassot, T. Connolly, and M. Diaz. Partial order quality of service to support multimedia connections: Reliable channels. In *2nd High Performance*

Distributed Computing Conference, Spokane, USA, July 1993.

[2] P. D. Amer, C. Chassot, T. Connolly, and M. Diaz. Partial order quality of service to support multimedia connections: Unreliable channels. In *International Networking Conference (INET)*, San Fransisco, USA, August 1993.

[3] P. D. Amer, P. T. Conrad, E. Golden, S. Iren, and A. Caro. Partially-ordered, partially-reliable transport service for multimedia applications. In *Advanced Telecommunications/Information Distribution Research Program (ATIRP) Conference*, pages 215–220, College Park, USA, January 1997.

[4] M. J. Andrews. XUDP: A real-time multimedia networking protocol. Bachelor thesis, Worcester Polytechnic Institute, March 1997.

[5] G. Barberis and D. Pazzaglia. Analysis and design of a packet-voice receiver. *IEEE Transactions on Communications*, 28(2):152–156, February 1981.

[6] V. Bhargava. Forward error correction schemes for digital communications. *IEEE Communications Magazine*, 21:11–19, January 1983.

[7] R. Braden, D. Clark, and S. Shenker. Integrated services in the internet architecture. RFC 1633, IETF, June 1994.

[8] Z. Chen, S-M. Tan, R. H. Campbell, and Y. Li. Real time video and audio in the world wide web. In *4th International World Wide Web Conference (WWW)*, Massachusetts, USA, December 1995.

[9] D. Clark and D. Tennenhouse. Architectual considerations for a new generation of protocols. *Computer Communication Review (ACM SIGCOMM)*, pages 200–208, September 1990.

[10] P. T. Conrad, E. Golden, P. D. Amer, and R. Marasli. A multimedia document retrieval system using partially-ordered/partially-reliable transport service. In *Multimedia Computing and Networking*, San Jose, USA, January 1996.

[11] B. Dempsey. *Retransmission-Based Error Control for Continuous Media Traffic in Packet-Switched Networks*. PhD thesis, University of Virginia, May 1994.

[12] B. Dempsey, T. Strayer, and A. Weaver. Adaptive error control for multimedia data transfers. In *International Workshop on Advanced Communications and Applications for High-Speed Networks (IWACA)*, pages 279–289, Munich, Germany, March 1992.

[13] B. J. Dempsey, J. Liebeherr, and A. C. Weaver. A new error control scheme for packetized voice over high-speed local area networks. In *18th Conference on Local Computer Networks (LCN)*, pages 91–100, Minneapolis, USA, September 1993.

[14] M. Diaz, K. Drira, A. Lozes, and C. Chassot. On the definition and representation of the quality of service for multimedia systems. In *International Conference on High Performance Networking (HPN)*, Spain, September 1995.

[15] N. G. Feamster. Adaptive delivery of real-time streaming video. Master's thesis, Massachusetts Institute of Technology, May 2001.

[16] F. Gong and G. Parulkar. An application-oriented error control scheme for high-speed networks. Technical Report WUCS-92-37, Washington University, November 1992.

[17] K-J Grinnemo and A. Brunstrom. Enhancing TCP for applications with soft real-time constraints. In *SPIE Multimedia Systems and Applications*, pages 18–31, Denver, USA, August 2001.

[18] K-J Grinnemo and A. Brunstrom. Evaluation of the QoS offered by PRTP-ECN – a TCP-compliant partially reliable transport protocol. In *9th International Workshop on Quality of Service (IWQoS)*, pages 217–231, Karlsruhe, Germany, June 2001.

[19] K. Hess. Media streaming protocol: An adaptive protocol for the delivery of audio and video over the Internet. Master's thesis, University of Illinois at Urbana-Champaign, 1998.

[20] S. Jacobs and A. Eleftheriadis. Streaming video using dynamic rate shaping and TCP congestion control. *Journal of Visual Communication and Image Representation*, 9(3), 1998.

[21] J-R Li, D. Dwyer, and V. Bharghavan. A transport protocol for heterogeneous packet flows. In *The 18th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, New York, USA, March 1999.

[22] R. Marasli, P. D. Amer, and P. T. Conrad. Retransmission-based partially reliable transport service: An analytic model. In *The 15th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 621–629, San Fransisco, USA, March 1996.

[23] R. Marasli, P. D. Amer, and P. T. Conrad. Partially reliable transport service. In *2nd Symposium on Computers and Communications (ISCC)*, Alexandria, Egypt, July 1997.

[24] B. Mukherjee and T. Brecht. Time-lined TCP for the TCP-friendly delivery of streaming media. In *IEEE International Conference on Network Protocols (ICNP)*, pages 165–176, Osaka, Japan, November 2000.

[25] C. Papadopoulos. *Error Control for Continuous Media and Large Scale Multicast Applications*. PhD thesis, Washington University, August 1999.

[26] C. Papadopoulos and G. Parulkar. Retransmission-based error control for continuous media applications. In *6th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, pages 5–12, Zushi, Japan, April 1996.

[27] M. Piecuch, K. French, G. Oprica, and M. Claypool. A selective retransmission protocol for multimedia on the Internet. In *SPIE Multimedia Systems and Applications*, Boston, MA, USA, November 2000.

[28] J. Postel. User datagram protocol. RFC 768, IETF, August 1980.

[29] J. Postel. Transmission control protocol. RFC 793, IETF, September 1981.

[30] K. Ramakrishnan and S. Floyd. A proposal to add explicit congestion notification (ECN) to IP. RFC 2481, IETF, January 1999.

[31] H. Schulzrinne. RTP: A transport protocol for real-time applications. RFC 1889, IETF, January 1996.

[32] B. C. Smith. Cyclic-UDP: A priority-driven best effort protocol. Unpublished, May 1994.

[33] R. Stewart, M. Ramalho, Q. Xie, M. Tuexen, and P. T. Conrad. SCTP partial reliability extension. Internet draft, IETF, May 2002. Work in Progress.

[34] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson. Stream control transmission protocol. RFC 2960, IETF, October 2000.

[35] J. Stone and C. Partridge. When the CRC and TCP checksums disagree. In *Computer Communication Review (ACM SIGCOMM)*, pages 309–319, Stockholm, Sweden, August 2000.

[36] W. Strayer, B. Dempsey, and A. Weaver. *XTP: The Xpress Transfer Protocol*. Addison-Wesley Publishing, July 1992.

[37] Q. Xie, R. Stewart, C. Sharp, and I. Rytina. SCTP unreliable data mode extension. Internet draft, IETF, April 2001. Work in Progress.