RTMC: An Error Control Protocol for IP-based Audio-Visual Multicast Applications

Georg Carle [1]
GMD Fokus, Berlin
carle@fokus.gmd.de

Jörg Ottensmeyer
Siemens AG, Corporate Technology, Munich
joerg.ottensmeyer@mchp.siemens.de

Abstract

Transport of Audio Visual Data is becoming an important application of the IP-based best-effort service in the Internet. We present an error control protocol for provision of a real-time reliable multicast transport service. The protocol RTMC (Real-Time MultiCast) uses Forward Error Correction (FEC) for protecting first transmissions (i.e., proactive error control) and retransmissions. Results of the performance evaluation show that RTMC error control imposes low latency, and is scalable to a large number of receivers.

1 Motivation

In the past, a widespread believe has been that *retransmission-based* error control is not suitable for interactive audio-visual applications. While the development of retransmission-based error control mechanisms for continuous media (CM) applications has been a research topic for several years, it was only recently that this topic became of widespread interest, leading to a significant number of improvements. This includes a large number of proprietary protocols that are currently being introduced as part of CM Internet applications, and as Web browser plug-ins. This motivated our design goal of realizing an implementation concept of downloadable protocol modules.

While the first generation of CM applications that deploy error control has been limited to simple point-to-point communication (unicast), recent developments also use multicast network services. As more and more end users get access to IP multicast services, error control schemes that allow the successful provision of multicast audio-visual services over the Internet are of significant economic importance.

In order to meet the reliability requirements of a real-time application, it is possible to use a network service that directly provides the required reliability, without additional error control mechanisms in the transport layer. This can be ensured by reservation of network resources, or by dimensioning the network in a way that the residual error probability is sufficiently small (over-engineering). In situations where a network service does not meet the reliability requirements of the application directly, additional error control mechanisms are required. By providing sufficient transport protocol processing capability with a low latency, it is possible to meet delay requirements of many audio-visual applications even after one or two retransmissions. This strategy potentially offers better utilization of network resources. It is particularly suitable for highly bursty traffic, as it allows to increase the load of intermediate systems up to a level in which losses are relatively frequent. As loss rates and loss characteristics may vary significantly, dimensioning error control parameters for proactive FEC and for retransmission needs to take into account the network scenario and the application scenario.

IP best effort service provides no guarantees for loss rate, delay, and in-sequence delivery. This service model is based on the hope that all traffic sources are well behaved. The congestion control algorithm of TCP ensures such a behaviour. However, as more and more open-loop applications based on UDP are used, higher losses can be observed, and the future of this service model is seriously threatened. There exist a significant number of publications that investigate the loss characteristic of the current Internet. IP losses for unicast communication are covered by Bolot [Bolo93] and Paxson [Paxs97]. IP losses for multicast communication over the MBONE are analyzed by Yajnik [YaKT96] and by Handley [Hand97]. The measurements of [YaKT96] showed a relatively high loss probability in the access area and rather low loss probabilities in the backbone area. In such loss scenarios,

[1] Part of this work has been done while Georg Carle was with Institut Eurecom, Sophia Antipolis, France

error control schemes are attractive that involve servers, or that apply local recovery. However, certain backbone links like the perpetually congested US/UK link may cause high losses in the backbone area, limiting the effectiveness of these error recovery schemes.
In the near future, Internet service providers will support different service classes, still offering the same service type 'best-effort-service', but with different service qualities (i.e., delay characteristic, loss rate and characteristic). In a future Internet, the widespread use of reservation protocols, such as RSVP, can be expected. In combination with access control techniques and scheduling mechanisms in all network nodes, reservation allows the provisioning of IP services with guaranteed quality of service. However, such networks will need some kind of tariffing to make the use of a service with guaranteed QoS more expensive than the use of a service without QoS guarantees. Therefore, users still will be motivated to apply powerful error control mechanisms in order to be able to use an inexpensive service class. In order to guarantee the required quality, *all* nodes from the transmitter to the receiver need mechanisms to support QoS guarantees. As IP services with guaranteed QoS will not be ubiquitously available for a relatively long time, powerful error control mechanisms will also continue to play an important role in the future.

2 Error Recovery with ARQ and FEC

2.1 ARQ for Continuos Media Streams

For transmission over the Internet, where delay and jitter are frequently in the order of a few hundred milliseconds, the support of *interactive* voice transmission using retransmission-based protocols (i.e. with Automatic Repeat reQuest mechanisms) has not yet been demonstrated. Considering the fact that interactive voice applications require round-trip delays of less than 200 ms [Klem67], retransmission generally seems not a feasible option. However, the situation is different for *distribution* of voice over the Internet. Recently, Xu, Myers, Zhang and Yavatkar [XuMZ97] have investigated the use of retransmission for the delivery of non-interactive voice over the Internet to multiple recipients. Given a playout delay in the order of 500 ms, retransmission for loss recovery is feasible. The authors argue that in the case of non-interactive voice, the receivers can make a trade-off between increased reliability and lower latency by choosing the playout delay appropriately.
Li, Paul, Pancha and Ammar [LiPA97] have recently proposed a retransmission-based loss recovery protocol, called Layered Video Multicast with Retransmission (**LVMR**), for non-interactive transmission of MPEG video to multiple receivers across the Internet. The MPEG video stream is separated into three layers: The base layer contains I-frames only, the other two layers contain P- and B-frames respectively. When the receiver detects a loss of a frame, he can send a NAK to request the retransmission. A retransmission is only requested if the data is likely to be received before it is required for playout. The recovery time depends on the round-trip time between receiver and the node that retransmits the frame and the processing times at reception. For LVMR loss recovery is local: Each receiver has a **designated receiver**, to whom the receiver will send the NAK to. The NAKs and the retransmissions are done via unicast to keep the overhead due to loss recovery low. The experiments performed for a playout delay of at least 1500 ms indicate that LVMR the is able to recover around 80 % of the losses seen by a receiver.

2.2 FEC for Continuos Media Streams

Today, an important limit for widespread use of telephony and video-conferencing over the Internet is bad service quality due to losses in congested routers. A number of interactive applications employ FEC for real-time error control with stringent delay requirements. As IP services may already have significant one-way-trip-times due to queuing in routers, special care is required in the development of the FEC scheme in order to obtain acceptable delay properties.
For audio applications that apply codecs with relatively long sampling intervals (e.g. as used for GSM), the resulting data stream has a relatively low packet rate, further complicating the design of a suitable FEC scheme. A number of applications were developed that have an application-specific FEC scheme with good delay properties.
The INRIA freephone [BoVe97] achieves good delay properties by encoding the audio stream using two different coding standards, and by transmitting encoded samples of the same time interval in subsequent packets: The data stream of freephone contains in every packet a PCM-encoded sample of one time interval, together with a redundant version of the *previous* time interval encoded at a lower rate. This FEC scheme has the advantages of adding only little bandwidth overhead to a PCM encoded audio stream, and of not increasing the IP packet rate while achieving relatively high robustness against losses.
An example for a video-specific FEC scheme is the Priority Encoding Transmission (**PET**) developed at ICSI, Berkeley [AlBE96]. This technique allows a user to specify a different priority for each segment of a continuous media stream. According to the assigned priority, PET generates a different amount of redundancy for the segments and disperses user data and redundancy onto several subsequent packets. PET can be applied to the transmission of MPEG video streams such that the data of one GOP is dispersed over a sequence of packets and that I-Frames are protected with a higher amount of

redundancy than P-Frames, which are protected by a higher amount of redundancy than B-Frames. A typical dimensioning would be to protect I-, P- and B-Frames with 100%, 33%, and 5% redundancy, respectively, and to disperse the data of a GOP over packets of 2 Kbyte length. PET was integrated into the MBONE video conferencing tool vic and was shown to work in combination with MPEG-1 and MPEG-2 data streams.

The Real-Time Transport Protocol (**RTP**, [ScCF96]) is a transport layer protocol framework which has been developed by the Internet Engineering Task Force (IETF) Audio/Video Transport working group in order to support delivery of continuous media over the Internet. The protocol defines a data packet semantic with timing information, packet sequence numbers, and optional parameters. Various payload formats are defined for different audio and video compression standards. No specific error control mechanism is defined. However, it is possible to adapt the framework specified by RTP by defining application-specific error control mechanisms. Current contributions describe how RTP can be combined with FEC [RoSc97]. Additionally, it has been proposed to combine RTP with the receiver-initiated retransmission scheme from SRM [Parn96]. The RTP framework also defines a control protocol RTCP (Real Time Control Protocol) which allows to collect feedback from the receivers. RTCP also can be adapted to application-specific needs. RTP is not only widely incorporated into Internet applications, but also adopted by ITU for the H.323 recommendation defining audio-visual telephone systems for local area networks.

3 RTMC Protocol Design

RTMC design principles are proactive FEC with Reed-Solomon codes, support for Application Data Units (ADUs) with different importance, and NAK-based retransmissions with suppression of unnecessary retransmissions (retransmissions arriving too late, and duplicated retransmissions).

In a proactive FEC scheme, data packets transmitted for the first time are protected by redundant packets. In order to provide for error control as long as are within their delay budget, and in order to stop error control for those outside the delay budget, the RTMC protocol provides timing functionality for both sender and receiver. Before a receiver requests a retransmission, and before a sender issues a retransmission, time bounds are calculated that compare the current delay budget with the best case retransmission time.

3.1 Receiver Mechanisms

Using redundant packets that are transmitted together with originally transmitted data packets, receivers try to reconstruct corrupted ADUs.

If reconstruction by FEC is not possible because the number of packets of a transmission group is not sufficient, a retransmission request is sent as long as the receiver does not estimate that the retransmission arrives too late. The calculation takes into account RTT and processing time. As Reed-Solomon-Codes are used, processing time increases with the number of redundancy packets of a transmission group (TG) of original data, and also with the number of original packets within a TG [Riz97].

3.2 Sender Mechanisms

In a multicast ARQ scheme, the sender has to avoid duplicated retransmission, initiated by identical NAKs from different receivers. To avoid duplicate retransmission, the receiver keeps a time stamp for every PDU that has been retransmitted. If the sender gets a NAK for the same PDU after having retransmitted, it has to check the following (It is assumed, that the first NAK arriving comes from the closest receiver sending this NAK):

- A NAK for a PDU that has already be retransmitted once has to be answered if it is actually the second retransmission request from a receiver for the same PDU. For this case the RTT of the receiver has to be examined.
- The NAK will not be answered if the request comes from a receiver that has a larger or the same RTT, as it is assumed that it already received an earlier retransmission.

The sender keeps a record for every retransmitted PDUs. This record contains a timestamp of the retransmission and an RTT value of the receiver that requested that frame. With the help of these information the sender can check if the NAK should be answered. For simplification, RTT values are ranges of RTT instead of precise RTT values.

The sender also has to avoid late retransmissions. Normally a retransmission request coming from one of the receivers could be answered without any further investigation. But queuing delays may delay a retransmission request so it arrives too late, in which case retransmission may not be performed. The bound on the sender side does not have to be as tight as the one on the receiver side.

As receivers check the remaining delay before issuing a NAK, the sender only discards NAK that have been delayed significantly.

To ensure first transmissions of PDUs in time to receivers that do not observe losses, the sender has to implement a

scheduling strategy. This scheduling strategy prefers the sending of a first transmission to the retransmission of an old PDU in cases where first transmissions are delayed by retransmissions more than a given threshold, by applying a rate control for retransmissions.

3.3 RTMC Protocol Data Units

The RTMC protocol uses the following protocol data units (PDUs): RTMC_Frame_PDUs that are segmented into RTMC_Segment_PDUs, RTMC_NAKs, and control messages (RTMC_CTRL_MSGs). The first two PDUs are explained in more detail.
Redundant information is added in form of additional RTMC_Segment_PDU. For Reed-Solomon-Encoding, the data is split up into groups of k segments. For every group of k data segments, a number of h redundant segments are constructed. The redundant segments will be sent after the original segments. In case where less than k segments are left to form a group, the number of redundant packets h will be calculated as a function of these k' remaining segments. This is shown in Figure 1.

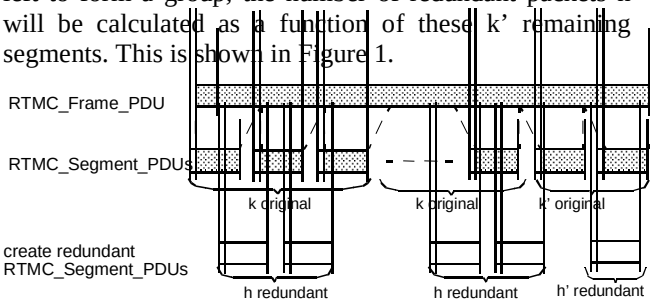

Figure 1: Segmentation using FEC Reed Solomon mechanism

The data format of an RTMC_Frame_PDU is shown in Figure 2. The following **SDU types** of it exist: Regular Data Frame (00); Acknowledgment-Frame (01); Fragment-Frame (10). The field **last-frag** indicates the last fragment in the case where a RTMC_Frame_PDU has to be fragmented. The field **redundancy** indicates the amount of redundancy that is used. The **FSN** field contains sequence number from 0...65535 of the RTMC_Frame_PDU. The **Length** field indicates the length of the RTMC_Frame_PDU payload. The **NextLength** field contains the length of the next frame to be sent. This information can be set or omitted (set to 0), since its use adds an additional delay of 1 frame. Usage is only recommended for high loss networks, where it supports recovery in cases where the first segment of a frame is lost.

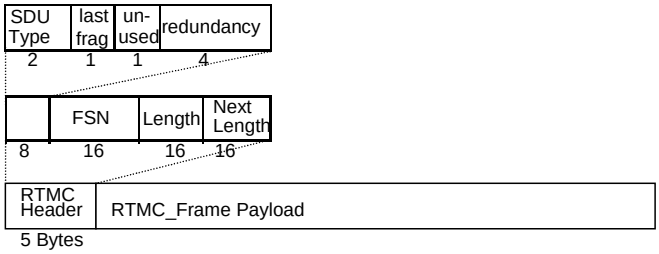

Figure 2: Data Structure of a RTMC_Frame_PDU

The RTMC_Segment_PDU contains a header field and a payload field as shown in Figure 3. The payload has a **constant** length throughout a session. An RTMC_Segment_PDU can be of one of the following **types**: Regular data segment (00); Redundant data segment (01); Regular retransmission data segment (10); Redundant retransmission data segment (11). If no segmentation is performed, the payload consists of an entire RTMC_Frame_PDU.

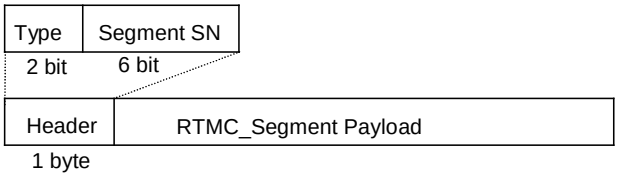

Figure 3: Data Structure of a RTMC_Segment_PDU

4 Performance Evaluation

In this section we present performance results of an RTMC protocol implementation as well as results of the effectiveness of the basic RTMC protocol mechanisms ARQ and FEC. Moreover, we asses the effectiveness of RTMC in heterogeneous multicast scenarios.

4.1 Overview on the RTMC Implementation

The RTMC protocol has been implemented using the protocol operating system CHANNELS [Boe95] which has been developed to support a platform independent implementation of modular communication protocols. The modules used in RTMC are shown in Figure 4. The main functionality of RTMC is realized in two modules. The control module is responsible for the initial configuration, for polling of debug information and for reconfiguration of redundancy parameters in the sender module. The sender module contains the functionality for segmentation and for FEC encoding while the receiver module contains the functionality for reassembly and FEC decoding.
The API module handles communication with the application. The Netglue modules handles IP multicast. The optional Error and Delay Module allows to emulate segment loss and additional delays for performance evaluation.

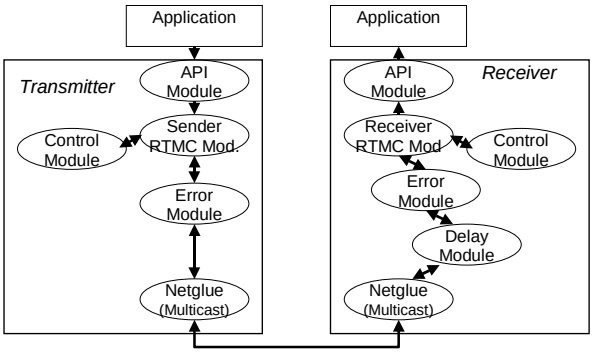

Figure 4: Modules of prototype RTMC implementation

First we discus the results for the FEC calculation at the transmitter side. For this analysis we traced the frame processing in our RTMC prototype. Frame processing time depends to a large extend on the sendbuffer, which also acts as the retransmission buffer. The call to update the sendbuffer costs $\tau_{sendbuffer}$ = 10 ms. For a fixed segment size which has the advantage of an optimized FEC calculation (we selected 100 bytes), a constant segment processing time $\tau_{segment}$ = 475 µs has been achieved in our implementation.

As shown by Rizzo in [Riz97], the processing cost for calculating the Reed-Solomon Code is proportional to the number of data segments K, to the number of redundant segments H, and to a processor-specific constant for encoding c_{enc}, i.e. $\tau_{calred} = K \cdot H \cdot c_{enc}$. For the hardware used for our implementation, we measured c_{enc} = 120 µs/kbyte.

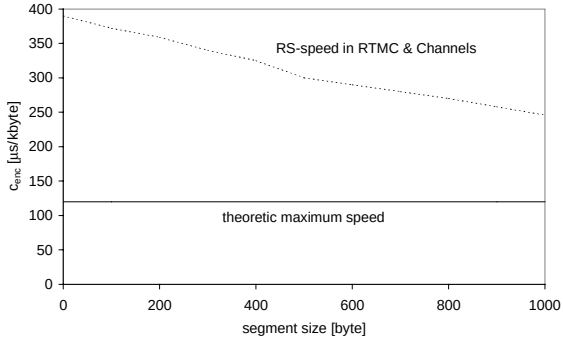

Figure 5: Time consumption for Reed-Solomon encoding

Figure 5 shows the measured speeds for different segment sizes. For all segment sizes the RTMC module is at least two times slower than the maximum speed of the encoder, caused by copy operations of the buffer management mechanisms.

With the above constants and parameters, the total time consumption per frame at the transmitter can be calculated by the following formula where R is the selected amount of redundancy in %.

$$\tau_{frame} = \tau_{sendbuffer} + \frac{size(frame)}{size(seg)} \bullet (\tau_{segment} + R \bullet (\tau_{calcred} + \tau_{segment}))$$

Figure 6 compares the delay calculated with the above formula and the measured delay. It turns out that the measured delay (solid lines) is close to the calculated delay (dashed lines). This indicates that the model for the total time consumption per frame used by the protocol for delay bound calculation is valid.

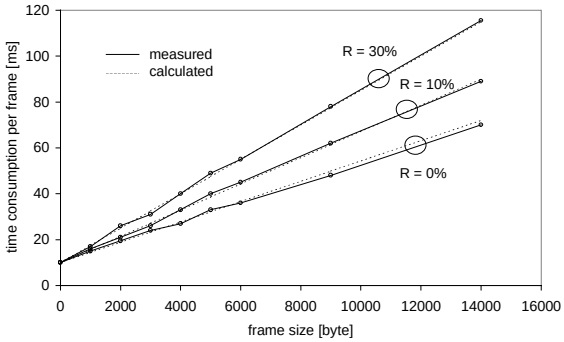

Figure 6: Time consumption for varying frame size and redundancy; comparing analytic model and measurements

Another important delay is the time consumption for a retransmission. In case a frame has to be retransmitted, the RTMC implementation checks the NAK to suppress duplicated retransmissions and late retransmission in order to send only the required retransmission. The time for sending a retransmission is equal to the time for sending a first transmission. Additional NAK processing is t_{nack} = 150 µs.

4.2 Assessment of RTMC mechanisms

For assessment of the RTMC mechanisms, we choose different scenarios that show the impact of the different error control mechanisms of the RTMC protocol. As performance measure we evaluated the remaining frame loss probability when transmitting a movie using RTMC. The selected movie "Thorax" is MPEG-1 encoded and has a length of 1356 frames. Its average size of I-frame is approximately 8 Kbytes, with most frames being significantly smaller.

4.2.1 Effectiveness of FEC

In the first test we focused on the FEC mechanism of the protocol. We transmitted the movie with different levels of redundancy and adjusted the segment loss rate between 10^{-5} and 10^{-1}. Figure 7 shows the obtained results.

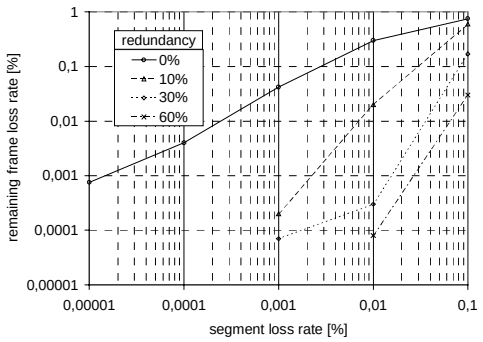

Figure 7: Measured frame loss probability for different amounts of redundancy

The following observations can be drawn. The remaining frame loss rate increases significantly with the segment loss rate. The frame loss rate decreases with the amount of redundancy. For a loss rate of less than 10^{-4} and redundancy >0% we did not observe any frame loss over the (short) playing time of the movie. However, using only FEC is not sufficient usually as temporary overload may cause higher loss rates than $10e^{-4}$.

4.2.2 Effectiveness of ARQ

In the second test we focused on the ARQ mechanism. We adjusted the segment loss rate between 10^{-5} and 10^{-1} and allowed for up to 3 retransmissions. Figure 8 shows the obtained results.

The remaining frame loss decreases with the number of retransmissions. In particular, it decreases by an order of magnitude with each additional retransmission allowed. Again, a frame loss rate below 10^{-4} can not be observed in a movie with 1500 frames.

4.2.3 Effectiveness of the hybrid scheme

In the third test we combined FEC and ARQ. We varied the segment loss rate between 10^{-5} and 10^{-1}. The test was performed with 20 receivers. For simplification we allowed for every receiver only one NAK per frame.

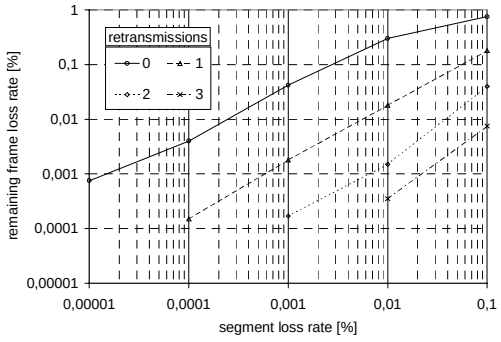

Figure 8: Measured remaining frame loss rate for 0 to 3 retransmissions

If the frame could not be reconstructed after a retransmission, no additional retransmission was requested even if the expected RTT would allow delivery within the planned playout time. The transmitter responds to a NAK with a retransmission to the multicast group of all segments belonging to this frame.

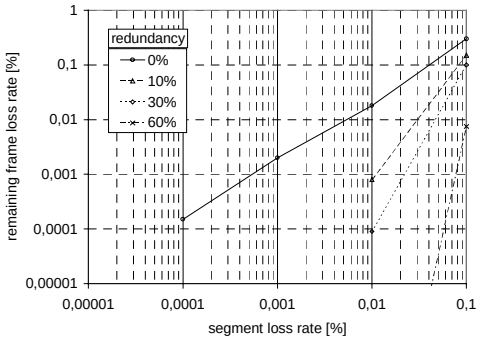

Figure 9: Measured remaining frame loss rate with different levels of redundancy and allowing only one retransmission per frame in a scenario with 20 receivers

The results of this test are shown in Figure 9. Again it can be seen that the remaining frame loss rate decreases with increasing redundancy. However, the remaining frame loss could be lowered significantly when compared with Figures 7 and 8. This clearly shows that a combination of both mechanisms is beneficial.

One question that arises with the use of a hybrid scheme is the additional overhead caused by redundancy and by retransmissions. Figure 10 evaluates the normalized data rate for this test.

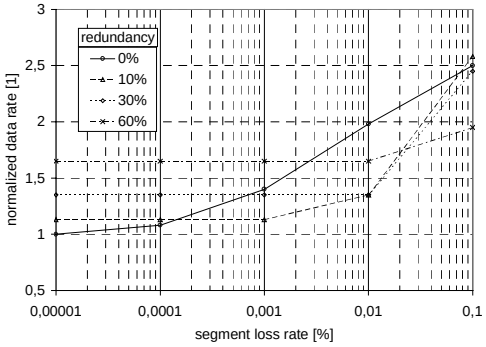

Figure 10: Measured normalized data rate for varying segment loss rates

The figure shows that, the lowest amount of redundancy caused the least number of retransmissions for low error rates. However, increasing loss rates lead to a rapidly increasing number of retransmission. A higher level of redundancy causes (unnecessary) higher load in case of low error rates, but saves bandwidth in case of higher error rates. This indicates that bandwidth requirements of the scheme can be optimised if the current error rate is

known, by selecting the appropriate amount of redundancy.

4.2.4 Effectiveness of RTMC in heterogeneous scenarios

For investigating of the effectiveness of RTMC, we selected the following two heterogeneous scenarios. In each scenario, heterogeneity is limited to a single parameter.

Different RTTs

In the first scenario (see Figure 11) loss mainly occurs on the shared link (loss rate 1%), with a smaller loss rate (0,3%) for each individual link. RTTs values range between 10 ms and 200 ms as shown in Figure 11.

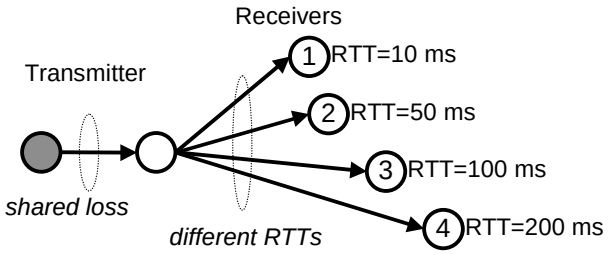

Figure 11: Test scenario with heterogeneous RTTs

An end-to-end delay budget of 250 ms leads to different numbers of possible retransmissions for each receiver. While receiver 4 can request only 1 retransmission, receiver 2 can request up to 4 retransmissions. Results for this scenario show that the remaining loss rate is almost the same for all receivers. It increases slightly with growing distance between transmitter and receiver. In this scenario, retransmissions are sent before the error is detected by receivers with large RTT. A receiver far away generates less NAKs than a receiver closer to the transmitter (receivers with large RTTs benefit form receivers with smaller RTTs).

	Recv. 1	Recv. 2	Recv. 3	Recv. 4
# NAKs	120	80	67	33
# frames lost	4 (0,3%)	5 (0,37%)	5 (0,37%)	6 (0,44%)

Different Losses

For the second heterogeneous scenario we used different loss rates on the individual links in combination with homogeneous RTTs as shown in Figure 12. The receiver with the highest loss rate determines the amount of error control.

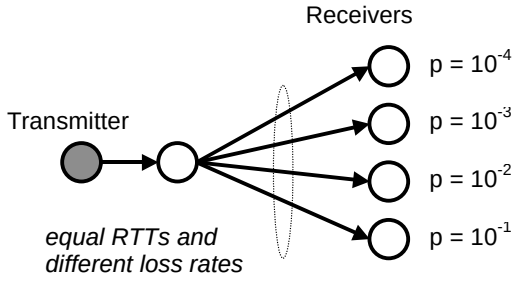

Figure 12: Heterogeneous RTT scenario

For the receiver with the smallest segment loss rate, no remaining frame loss has been observed for each level of redundancy. This receiver did send NAKs, and all retransmissions have been successful. Receivers with higher loss rates could not recover all corrupted frames. However, the remaining loss rate is significantly lower for an increasing amount of redundancy.

Redund.	Recv. 1	Recv. 2	Recv. 3	Recv. 4
0%	0	9 (0,7%)	22 (1,6%)	353 (26%)
10%	0	0	2 (0,15%)	198 (15%)
30%	0	0	1 (0,07%)	40 (3%)
60%	0	0	0,5 (0,04%)	10 (0,7%)

5 Conclusions

In this paper we described the design of an error-control protocol for audio-visual multicast applications and did a comprehensive performance evaluation of its prototype implementation. The following advantages of RTMC have been demonstrated:

- The FEC scheme of RTMC allows for low latency error control. Other schemes impose significanty larger latency, e.g. the FEC scheme of PETs is based on a whole MPEG GOP.
- The measurements showed that RTMC is suited for different scenarios. In particular, it adapts to heterogeneous receives and may even take advantage of heterogeneous RTTs.
- RTMC is scalable to a large number of receivers since many errors can be recovered using the proactive FEC, requiring no feedback.

However, RTMC has also some limitations:

- Since it is based on a closed loop scheme it exhibits an increased end-to-end delay. However, for a given end-to-end budget the proposed mechanisms allow to maximize the gain.
- Since RTMC uses retransmissions, there may be a significant delay variation between consecutive frames. The hybrid error control scheme therefore leads to significant playout buffer requirements. While this is not a problem for end systems such as PCs, it

may be a problem for systems with limited resources such as PDAs or set top boxes.

In summary, the results indicate a wide applicability of the RTMC protocol. Future technology enhancements (in particular CPU improvements) will increase the performance of a protocol like RTMC, and therefore will increase its applicability.

The implementation of RTMC is prepared for the concept of downloadable modules. For the prototype implementation we used the Channels environment. While the current version of Channels uses C++ a Java version will be available soon, enabling a quick and platform-independent deployment of RTMC.

Acknowledgements

The authors would like to thank Ernst Biersack and Stefan Böcking for valuable discussions, and Stephan Röösli for implementation and measurements. The support of Georg Carle by the European Commission in form of a TMR (Training and Mobility for Researchers in Europe) grant with Institute Eurecom is gratefully acknowledged.

References

[AlBE96] A. Albanese, J. Blömer, J. Edmonds, M. Luby, and M. Sudan, "Priority Encoding Transmission", IEEE Transactions on Information Theory (special issue devoted to coding theory), 42(6), November 1996.

[Boe95] S. Böcking: "Channels - A Run-Time System for Multimedia Protocols", In Proceedings of ICCCN 95, Sept. 95, see also: http://www.siemens.de/zfetsn2/inw/channels/

[Bolo93] J.-C. Bolot, "End-to-end Packet Delay and Loss Behavior in the Internet", In Proceedings of ACM SIGCOMM '93, pp. 289-298, San Francisco, CA, 1993.

[BoVe97] J. C. Bolot, A. Vega Garcia, "The case for FEC-based error control for packet audio in the Internet", ACM Multimedia Systems, 1997.

[CaBi97] G. Carle, E. Biersack: "Survey on Error Recovery for IP-based Audio-Visual Multicast Applications, IEEE Network Mag., Nov./Dec. 97, pp. 24-36.

[Hand97] M. Handley, "An Examination of MBONE Performance", Technical Report, Univ. of Southern California, Information Sciences Institute USC/ISI, ISI/RR-97-450, Jan. 1997.

[Klem67] E. Klemmer, "Subjective Evaluation of Transmission Delay in Telephone Conversations", Bell Systems Technical Journal, 46:1141-1147, July 1967.

[LiPA97] X. Li, S. Paul, P. Pancha, and M. Ammar, "Layered Video Multicast with Retransmission (LVMR): Evaluation of Error Recovery Schemes", In G. Parulkar, editor, NOSSDAV 97, Springer, May 1997.

[Parn96] P. Parnes, "RTP extension for Scalable Reliable Multicast; Internet-Draft: draft-parnes-rtp-ext-srm", Work in Progress, November 1996.

[Paxs97] V. Paxson, "End-to-End Internet Packet Dynamics", Computer Communication Review, Proceedings of ACM SIGCOMM'97 Conference, Cannes, France, September 1997, 27(4):139-152, October 1997.

[Riz97] L. Rizzo: "Effective Erasure Codes for Reliable Computer Communication Protocols", Computer Communication Review, 27(2), pp. 24-36, 4.97

[RoSc97] J. Rosenberg and H. Schulzrinne, "An A/V Profile Extension for Generic Forward Error Correction in RTP", Internet-Draft: draft-ietf-avt-fec (Work in Progress), IETF, July 1997.

[ScCF96] H. Schulzrinne, S. Casner, R. Frederic, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", Request for Comments RFC 1889, IETF, January 1996.

[XuMZ97] X. Xu, A. Myers, H. Zhang, and R. Yavatkar, "Resilient Multicast Support for Continuous-Media Applications", In G. Parulkar, editor, NOSSDAV 97, Springer, May 1997.

[YaKT96] M. Yajnik, J. Kurose, and D. Towsley, "Packet Loss Correlation in the MBone Multicast Network", In Proceedings of IEEE Global Internet, London, UK, November 1996, IEEE.