Internet Engineering Task Force
Internet Draft

Document: draft-ietf-avt-uxp-07.txt
October 2004

Expires: April 2005

G. Liebl
LNT, Munich Univ. of Technology

M. Wagner, J. Pandel, W. Weng
Siemens AG, Munich

An RTP Payload Format for Erasure-Resilient Transmission of Progressive Multimedia Streams

Status of this Memo

Copyright Notice

Abstract

This document specifies an efficient way to ensure erasure-resilient transmission of progressively encoded multimedia sources via RTP using Reed-Solomon (RS) codes together with interleaving. The level of erasure protection can be explicitly adapted to the importance of the respective parts in the source stream, thus allowing a graceful degradation of application quality with increasing packet loss rate on the network. Hence, this type of unequal erasure protection (UXP) schemes is intended

to cope with the rapidly varying channel conditions on wireless

access links to the Internet backbone. Furthermore, protection of non-progressive multimedia streams is ensured, since equal erasure protection (EXP) represents a subset of generic UXP. By applying interleaving and RS codes a payload format is defined, which can be easily integrated into the existing framework for RTP.

Table of Contents

1. Introduction

Due to the increasing popularity of high-quality multimedia applications over the Internet and the high level of public acceptance of existing mobile communication systems, there is a strong demand for a future combination of these two techniques: One possible scenario consists of an integrated communication environment, where users can set up multimedia connections anytime and anywhere via radio access links to the Internet. For this reason, several packet-oriented transmission modes like EGPRS (Enhanced General Packet Radio Service) or UMTS (Universal Mobile Telecommunications System) can be used, which are mostly based on the same principle: Long message blocks, i.e. IP packets, that enter the wireless part of the network are split up into segments of desired length, which can be multiplexed onto link layer packets of fixed size. The latter are then transmitted sequentially over the wireless link, reassembled, and passed on

to the next network element.
However, compared to the rather benign channel characteristics on today's fixed networks, wireless links suffer from severe fading, noise, and interference conditions in general, thus resulting in a comparably high residual bit error rate after detection and decoding. By use of efficient CRC-mechanisms, these bit errors are usually detected with very high probability, and every corrupted segment, i.e. which contains at least one erroneous

bit, is discarded to prevent error propagation through the network. But if only one single segment is missing at the reassembly stage, the upper layer IP packet cannot be reconstructed anymore. The result is a significant increase in packet loss rate at IP level.
Since most multimedia applications can only recover from a very limited number of lost IP packets, it is vitally necessary to keep packet loss at IP level within a certain acceptable range depending on the individual quality-of-service requirements. However, due to the delay constraints typically imposed by most audio or video codecs, the use of ARQ-schemes is often prohibited both at link level and at transport level. In addition, retransmission strategies cannot be applied to any broadcast or multicast scenarios. Thus, forward erasure correction strategies have to be considered, which provide a simple means to reconstruct the content of lost packets at the receiver from the redundancy that has been spread out over a certain number of consecutive packets.
There already exist some previous studies and proposals regarding erasure-resilient packet transmission [RFC2733,Hor99]. Since most of them are based on the assumption that all parts in a message block are equally important to the receiver, i.e. the respective application cannot operate on partly complete blocks, they were optimized with respect to assigning equal erasure protection over the whole message block. However, recent developments both in audio and video coding have introduced the notion of progressively encoded media streams, for which unequal erasure protection strategies seem to be more promising, as it will be explained in more detail below. Although the scheme defined in [RFC2733] is in principle capable of supporting some kind of unequal erasure protection, possible implementations seem to be quite complex with respect to the gain in performance. Finally, in [RFC2733] it is assumed that consecutive RTP packets can have variable length, which would cause significant segmentation overhead at the link layer of almost all wireless systems.

This document defines a payload format for RTP, such that different elements in a progressively encoded multimedia stream can be protected against packet erasures according to their respective quality-of-service requirement. The general principle, including the use of Reed-Solomon codes together with an appropriate interleaving scheme for adding redundancy, follows the ideas already presented in [Alb96], but allows for finer granularity in the structure of the progressive media stream. The proposed scheme is generic in the way that it (1) is independent of the type of media stream, be it audio or video, and (2) can be adapted to varying transmission quality very quickly by use of inband-signaling.

2. Conventions used in this Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119.

3. Preliminaries

The purpose of this section is to provide some preliminaries which are important for understanding the UXP scheme. First, some definitions used throughout this document are given. Next, Reed-Solomon Codes are introduced. Finally, progressive source coding and the resulting properties of progressive bitstreams are discussed.

3.1 Definitions

The following terms are used throughout this document:

1.) Segment: denotes a link layer transport unit.
2.) Segmentation/Reassembly Process: If the size of the transport units at the link layer is smaller than that at the upper layers, message blocks have to be split up into several parts, i.e. segments, which are then transmitted subsequently over the link. If nothing is lost, the original message block can be restored at the receiving entity (reassembly).
3.) Codec: denotes a functional pair consisting of a source encoding unit at the sender and a corresponding source decoding unit at the receiver; usually standardized for different media applications like audio or video.
4.) Media stream: A bitstream which results at the output of an encoder for a specific media type, e.g. H.263, MPEG-4 Visual.
5.) Progressive media stream: A media stream which can be divided into successive elements. The distinct elements are of different importance to the decoding process and are commonly ordered from highest to least importance, where the latter elements depend on the previous.
6.) Progressive source coding: results in a progressive media stream.
7.) Reed-Solomon (RS) code: belongs to the class of linear nonbinary block codes, and is uniquely specified by the block length n, the number of parity symbols t, and the symbol alphabet.
8.) n: is a variable, which denotes both the block length of a

RS codeword, and the number of columns in a TB (see 19).

9.) k: is a variable, which denotes the number of information symbols in an RS codeword.

10.) t: is a variable, which denotes the number of parity symbols in an RS codeword.
11.) Erasure: When a packet is lost during transmission, an erasure is said to have happened. Since the position of the erased packet in a sequence is usually known, a corresponding erasure marker can be set at the receiving entity.
12.) Base layer: comprises the first and most important elements of the progressive media stream, without which all subsequent information is useless.
13.) Enhancement layer: comprises one or more sets of the less important subsequent elements of the progressive media stream. A specific enhancement layer can be decoded, if and only if the base layer and all previous enhancement layer data (of higher importance) are available.
14.) Info stream: denotes the bitstream which has to be protected by the UXP scheme. It usually consists of the media stream (progressively source encoded or not), which is arranged according to a desired syntax (e.g. to achieve an appropriate framing, see Sect. 5.4). In any case, it is assumed that every info stream is already octet-aligned according to the standard procedures defined in the context of the used syntax specifications.
15.) Info octet: Denotes one element of the info stream.
16.) Transmission block (TB): denotes a memory array of L rows and n columns. Each row of a TB represents a RS codeword, whereas each column, together with the respective UXP header (see 36) in front, forms the payload of a single RTP packet. Each TB consists of at least two distinct transmission sub blocks (TSB, see20): The first L_s rows belong to the signaling TSB, whereas the last L_d=(L-L_s) rows belong to one or more data TSB.
17.) Transmission sub block (TSB): denotes a memory array of 0<l<L rows and n columns, which is a horizontal slice of a TB. Depending on whether the info octet positions are filled with descriptors (see31) or media data, the TSB is of type signaling or data, respectively.
18.) L: is a variable, which denotes both the number of rows in a TB and the payload length (without UXP header, see 36) of an RTP packet in octets.
19.) Unequal erasure protection (UXP): denotes a specific strategy which varies the level of erasure protection across a TB according to a given redundancy profile.
20.) Equal erasure protection (EXP): is a subset of UXP, for which the level of erasure protection is kept constant across a TB.
21.) Redundancy profile: describes the size of the different erasure protection classes in a TB, i.e. the number of rows

(codewords) per class.

22.) Erasure protection class: contains a set of rows (codewords) of the TB with same erasure correction capability.

23.) i: is a variable, which denotes the number of parity symbols for each row in erasure protection class i.

24.) EPC_i: is a variable, which denotes the set of rows contained in erasure protection class i.
25.) R_i: is a variable, which denotes the total number of rows contained in erasure protection class i, i.e. the cardinality of EPC_i.
26.) T: is a variable, which denotes the number of parity symbols for each row in the highest erasure protection class (with respect to application data) in a TB.
27.) EPV: denotes the erasure protection vector of length (T+1) used to describe a certain redundancy profile.
28.) DP: descriptor used for in-band signaling of the erasure protection vector.
29.) SI: stuffing indicator, which contains the number of media stuffing symbols at the end of a data TSB (see 34).
30.) Descriptor Stuffing: insertion of otherwise unused descriptor values (i.e. 0x00) at the end of the signaling TSB. Descriptor stuffing is performed, if the final sequence of descriptors and stuffing indicators for a valid redundancy profile is shorter than the space initially reserved for it in the signaling TSB.
31.) Media Stuffing: insertion of additional symbols at the end of a data TSB. Media stuffing is performed, if the info stream (see 17) is shorter than the space reserved for it in the data TSB for a desired redundancy profile. Since the number of stuffing symbols is signaled in the respective SI, any octet value may be used (e.g. 0x00).
32.) Interleaver: performs the spreading of a codeword, i.e. a row in the TB, over n successive packets, such that the probability of an erasure burst in a codeword is kept small.
33.) UXP header: is the additional header information contained in each RTP packet after UXP has been applied. It is always present at the start of the payload section of an RTP packet.
34.) X: denotes a currently not used extension field of 1 bit in the UXP header.
35.) P: is a variable which denotes the number of parity symbols per row used to protect the inband signaling of the redundancy profile.
36.) ceil(.): denotes the ceiling function, i.e. rounding up to the next integer.

3.2 Reed-Solomon Codes

Reed-Solomon (RS) codes are a special class of linear nonbinary block codes, which are known to offer maximum erasure correction capability with minimum amount of redundancy.

An arbitrary t-erasure-correcting (n,k) RS code defined over Galois field GF(q) has the following parameters [Lin83]:

- Block length: n=q-1
- No. of information symbols in a codeword: k
- No. of parity-check symbols in a codeword: n-k=t
- Minimum distance: d=t+1

In what follows, only systematic RS codes over GF(2^8) shall be considered, i.e. the symbols of interest can be directly related to a tuple of eight bits, which is commonly called an octet in packet transmission. The principle structure of a codeword is shown in Fig. 1.
By shortening the initial (n=255,n-t) RS code, any desired (n',n'-t) RS code for a given erasure correction capability t may be obtained.

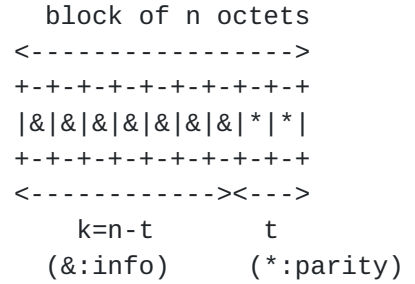


Fig. 1: Structure of a systematic RS codeword

3.3 Progressive Source Coding

The output of an encoder for a specific media type, e.g. H.263 or MPEG-4 Visual is said to be a media stream. If the media stream consists of several distinct elements, which are of different importance with respect to the quality of the decoding process at the receiver, then the media stream is progressive. The progressive media stream is often organized in separate layers. Hence, there exists at least one layer, often called base layer, without which decoding fails at all, whereas all the other layers, often called enhancement layers, just help to continually improve the quality. Consequently, the different layers are usually contained in the (source-)encoded media stream in decreasing order of importance, i.e. the base layer data is followed by the various enhancement layers.
An example can be found in the fine granular scalability modes which have been proposed to various standardization bodies like MPEG, where the resolution of the scaling process in the

progressive source encoder is as low as one symbol in the enhancement layer [Li01]. Another example is given by data partitioning which can be applied to the ITU/MPEG H.264/AVC

standard [Bla00], MPEG-4, and H.263++. Also, the existence of I,P, and B frames in streams which comply with standards like MPEG-2 can be interpreted as progressive.
From the above definition, it is quite obvious that the most important base layer data must be protected as strongly as possible against packet loss during transmission. However, the protection of the enhancement layers can be continually lowered, since a loss at these stages has only minor consequences for the decoding process. Thus, by using a suitable unequal erasure protection strategy across a progressive media stream, the overhead due to redundancy is reduced. Furthermore, if channel conditions get worse during transmission (resulting in a higher number of corrupt segments and thus higher IP packet loss rate), only more and more enhancement layers are lost, i.e. a graceful degradation in application quality at the receiver is achieved [Bur99].
Nevertheless, it should be mentioned that the specific structure of the media stream strongly depends on the actual media codec in use and does not always provide suitable mechanisms for transport over data networks, like framing (see also Sect. 5.4). In order to keep the description of the unequal erasure protection strategy in Sect. 4 as general as possible, the final bitstream which has to be protected by the proposed UXP scheme will be called "info stream" in the following. Furthermore, it is assumed that every info stream is already octet-aligned according to the standard procedures defined in the context of the used syntax specifications.

4. General UXP Concept

In this section, the principle features of the proposed UXP scheme are described with a special focus on the protection and reconstruction procedure which is applied to the info stream. In addition, the behavior of the sender and receiver is specified as far as it concerns the reconstruction of the info stream.
However, the complete UXP payload structure, including the additional UXP header, is described in Sect. 5.
The reason for using the term "info stream", as well as the details of the construction, are described in Sect. 5.4 . For now, we assume that we have an info stream which has to be protected.

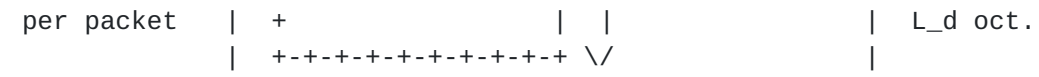
4.1 Transmission Block Structure

Fig. 1 already illustrated the structure of a systematic RS

codeword, which shall be represented by a single row with n successive symbols that contain the information and the parity octets. This structure shall now be extended by forming a transmission block (TB) consisting of L codewords of length n

octets each, which amounts to a total of L rows and n columns [Lie99]: Each column, together with the respective UXP header in front, shall represent the payload of an RTP packet, i.e. the whole data of a TB is transmitted via a sequence of n RTP packets all carrying a payload of length (L+2) octets (UXP header included).
Each TB usually consists of two or more horizontal sub blocks, the so-called transmission sub blocks (TSB), as can be seen in Fig. 2: The first L_s rows always belong to the signaling TSB, which is used to convey the actual redundancy profile in the data part to the receiver (see 5.5.). The following L_d=(L-L_s) rows belong to one or more data TSBs, which contain the interleaved and RS encoded info stream, as will be described below.

Transmission Block (TB)


per packet
L_d oct.

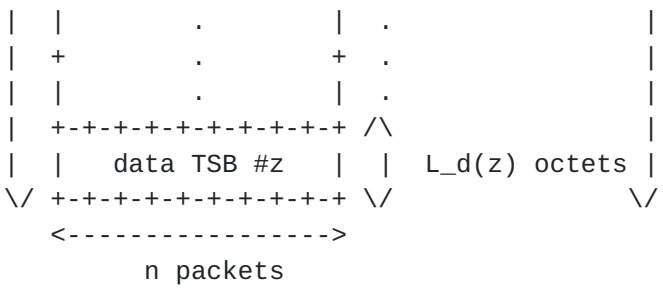


Fig. 2: General structure of a TB

Since the UXP procedure is mainly applied to the data TSBs, it will be described next, whereas the content and syntax of the signaling TSB will be defined in section 5.5.

4.2 TB Fill Procedure

For means of simplification, only one single data TSB will be assumed throughout the following explanation of the encoding and decoding procedure. However, an extension to more than one data TSB per TB is straightforward, and will be shown in section 5.6. In the following description, we need an info stream which is filled into a TSB. In order to make clear how the filling works in detail, we denote the octets of a stream as described in Fig. 3.

```
Octet pos: 0  1  2  3 ...              10            15
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
Octet    |x0|x1|x2|x3|x4|x5|x6|x7|x8|x9|xA|xB|xC|xD|xE|xF|

Octet pos:16             ...               ..        31
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
Octet    |xG|xH|.................................|xU|xV|

Octet pos: 32                                  44
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--|
Octet    |xW|xX|xY|xZ|y0|y1|.................y8|
```

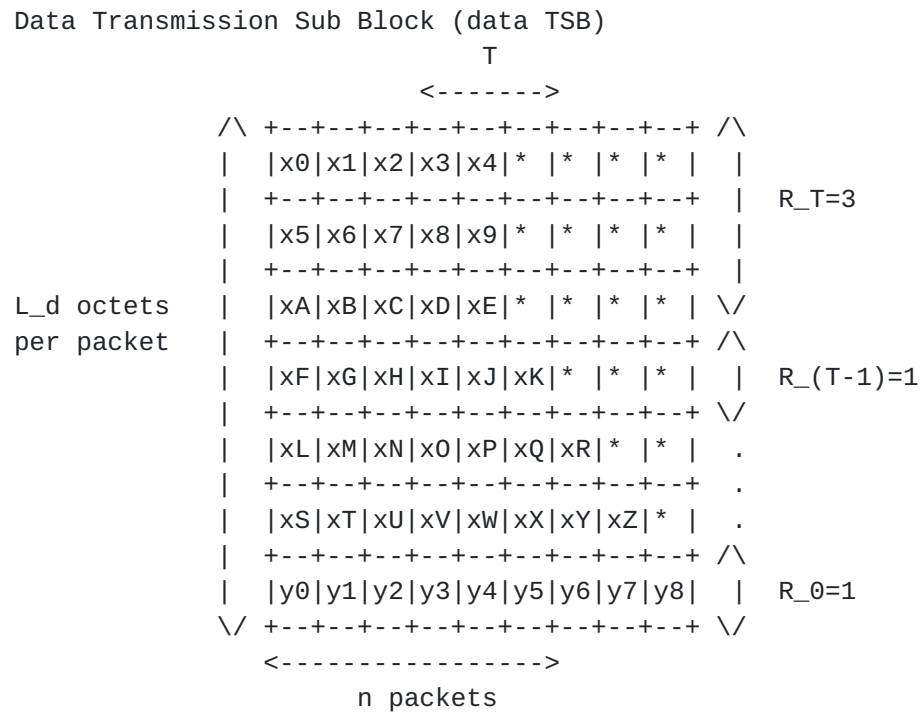
Figure 3: Exemplary info stream

This means, for example, that the octet at position 10 in the info stream is denoted by xA. The info stream is progressive, which means that the octets at the beginning of the stream are more important than the octets later in the stream.

As depicted in Fig. 4, the rows of a transmission sub block shall

be assembled into T+1 different classes EPC_i, where i=0...T,
such that each class contains exactly R_i=|EPC_i| consecutive

rows of the matrix, where the R_i have to satisfy the following relationship:
R_0+R_1+...+R_T=L_d



x#,y# : info octets belonging to the info stream defined in Fig. 3
* : parity octets gained from Reed-Solomon coding

Fig. 4: General structure for coding with unequal erasure protection

Furthermore, all rows in a particular class EPC_i shall contain exactly the same number of parity octets, which is equal to the index i of the class. For each row in a certain class EPC_i, the same (n,n-i) RS code shall be applied.
As can be observed from Fig. 4, class EPC_T contains the largest

number of parity octets per row, i.e. offers the highest erasure protection capability in the block. Consequently, the most important elements in the info stream must be assigned to class

EPC_T, where the value of T should be chosen according to the desired outage threshold of the application given a certain packet erasure rate on the link.
All other classes EPC_(T-1)...EPC_0 shall be sequentially filled with the remaining elements of the info stream in decreasing order of importance as follows: The info stream is filled into the TSB column by column, from left to right, and line by line, from the upper lines to the lowest line. The result of this procedure is shown in Fig 4.

In the following, we describe a set of rules containing a compact description of all the operations that must be performed for each transmission block at the sender and receiver.

4.3 UXP Sender Rules

1) The total number of columns n of the TB shall be chosen according to the actual delay constraints of the application.
2) The maximum erasure correction capability T and the R_i in the data TSB should be chosen according to the desired outage threshold of the application given the actual packet erasure rate on the link and the properties of the info streams. However, the resulting number of TSB rows, L_d=R_0+R_1+...+R_T, should be kept in mind since it has major influence on the packet size of the resulting RTP packets (cf. Sec.55555555 55).
3) Any suitable optimization algorithm may be used for deriving adequate values for T and all R_i. However, the result has to satisfy the following constraints:
   a. All available info octet positions in the data TSB have to be completely filled. If the info stream is too short for a desired profile, media stuffing may be applied to the empty info octet positions at the end of the data TSB by appending a sufficient number of stuffing octets. The stuffing octets MUST have the value 0x00. The actual number of stuffing symbols per data TSB is then signaled via the respective stuffing indicator (see Sect. 5.5.).
   b. The info stream SHOULD be fully contained within the data TSB (unless cutting it off at a specific point is explicitly allowed by the properties of the info stream).
4) For each nonempty class EPC_i, i=T...0, in the data TSB, the following steps have to be performed:
   a. All rows of this specific class SHALL be filled from left to right and top to bottom with data octets of the info stream as shown in Fig. 4.
   b. For each row in the class, the required i parity-check octets are computed from the same set of codewords of an

(n,n-i) RS code, and filled in the empty positions at the end of each row. Thus, every row in the class constitutes a valid codeword of the chosen RS code.

5) After having filled the whole data TSB with information and parity octets, the redundancy profile is mapped to the signaling TSB as described in section 5.5.
6) Each column of the resulting TB is now read out octet-wise from top to bottom and, together with the respective UXP header (see section 5.2) in front, is mapped onto the payload section of one and only one RTP packet.
7) The n resulting RTP packets SHALL be transmitted consecutively to the remote host, starting with the leftmost one.

4.4 UXP Receiver Rules

1) At the corresponding protocol entity at the remote host, the payload (without the UXP header) of all successfully received RTP packets belonging to the same sending TB SHALL be filled into a similar receiving TB column-wise from top to bottom and left to right.
2) For every erased packet of a received TB, the respective column in the TB SHALL be filled with a suitable erasure marker.
3) Before any other operations can be performed, the redundancy profile MUST be restored from the signaling TSB according to the procedure defined in Sect. 5.5.. If the attempt fails because of too many lost packets, the whole TB SHALL be discarded and the receiving entity should wait for the next incoming TB.
4) If the attempt to recover the redundancy profile has been successful, a decoding operation SHALL be performed for each row of the data TSB by applying any suitable algorithm for erasure decoding.
5) For all rows of the data TSB for which the decoding operation has been successful, the reconstructed data octets are read out from left to right and top to bottom, and appended to the reconstructed version of the info stream.

4.5 Protection Properties of UXP

One can easily realize that the above rules describe an interleaved coding scheme, i.e. at the sender a single codeword of a TB is spread out over n successive packets. Thus, each codeword of a transmitted TB experiences the same number of erasures at exactly the same positions.
Two important conclusions can be drawn from this:
a) Since the same RS code is applied to all rows contained in a specific class, either all of them can be correctly decoded or

none. Hence, there exist no partly decodable classes at the receiver.

b) If decoding is successful for a certain class EPC_i, all the classes EPC_(i+1)...EPC_T can also be decoded, since they are

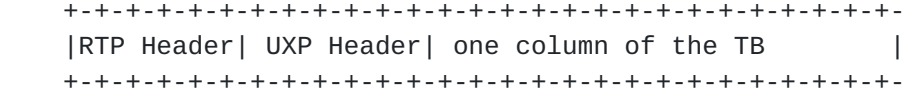
protected by at least one more parity octet per row. Together with rule 6, it is therefore always ensured, that in case a decodable enhancement layer exists, all other layers it depends on can also be reconstructed!

4.6 Description of the Redundancy Profile by Erasure Protection Vectors

Given the maximum erasure protection value T, the redundancy profile for a data TSB of size (L_d x n) SHALL be denoted by a so-called erasure protection vector EPV of length (T+1), where
EPV:=(R_0,R_1,...,R_(T-1),R_T)
From the above definition, it is easy to realize that the trivial cases of no erasure protection and EXP are a subset of UXP:
a) no erasure protection at all: all application data is mapped onto
   class EPC_0, i.e. EPV=(L_d,0,0,...,0).
b) EXP: all application data is mapped onto class EPC_T, i.e.
   EPV=(0,0,...,0,R_T=L_d).
Hence, the UXP payload format can also be used with info streams which are non progressive.

5. RTP payload structure

This section is organized as follows: First, the specific settings in the RTP header are shown. Next, the RTP payload header for UXP (the so-called UXP header) is specified. After that, the structure of the bitstream which is protected by UXP, the so-called info stream, is discussed. Finally, the in-band signaling of the erasure protection vector is introduced.
For every packet, the UXP payload is formed by reading out a column of the TB and prefixing it with the UXP header. Thus, a UXP-compliant RTP packet looks as follows:



5.1 Specific Settings in the RTP Header

The timestamp of each RTP packet SHALL be set to the sampling timestamp of the first octet of the progressive media stream in the corresponding TB. The clock rate MUST be the same as defined

in the RTP payload format for the progressive media stream.
If several data TSBs are included in one TB, the sampling
timestamp of data TSB #1 SHALL be relevant. This results in the

TS value being the same for all RTP packets belonging to a specific TB.
The payload type SHALL be of dynamic type, and obtained through out-of-band signaling similar to [RFC2733]. End systems, which cannot recognize a payload type, MUST discard it.
The marker bit SHOULD be set to 1 in the last packet of a TB; otherwise, its value SHOULD be 0.

5.2 Structure of the UXP Header

The UXP header SHALL consist of 2 octets, and is shown in Fig. 5:

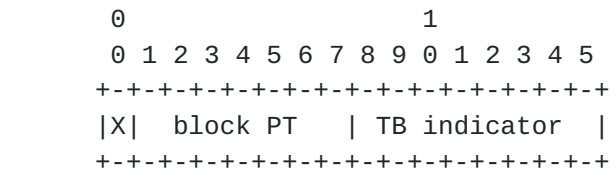


Fig. 5: Proposed UXP header

The fields in the UXP header are defined as follows:
- X (bit 0): extension bit, reserved for future enhancements, currently not in use -> default value: 0
- block PT (bits 1-7): regular RTP payload type to indicate the media type contained in the info stream
- TB indicator (bits 8-15): This field indicates the size and position of one TB within a stream of RTP-packets. The interpretation depends on the actual RTP sequence number of this packet. We denote the TB this packet belongs to as the current TB. Then there are two cases:
    1) If the sequence number is even, it indicates the total number of RTP packets within the current TB (which equals the number of columns of the current TB).
    2) Otherwise, it indicates the sequence number of the first RTP packet of the current TB. Since it is only one octet, it contains the least significant octet of the sequence number.

The syntax of the info stream which is protected by UXP is specified by the RTP payload type field contained in the UXP header. The details of the info stream are described in Sec. 5.4 .

Based on the RTP sequence number, the marker bit, and the TB indicator in each UXP header, the receiving entity is able to recognize both TB boundaries and the actual position of packets

(both received and lost ones) in the TB. An example how this can be done is given in the next subsection.

5.3 Usage of UXP- and RTP-Header at the Receiver

This subsection describes how the UXP- and RTP-headers can be used to reconstruct a TB.
We assume that the receiver knows about the sequence number of the first RTP packet within a TB, i.e. the left column, and the width of the TB. Then it is easy to find out the column in which the payload of an RTP packet has to be inserted only by considering the RTP sequence number.
However, the receiver does not know this in advance, since the TB width can be changed each time a new TB is sent. In addition, the RTP session starts with a random sequence number. Therefore, even if the TB width is known at the beginning, the receiver does not know whether the first packets where lost or not. It is then wrong to interpret the first received packet as the first packet in the TB.
Therefore, the comination of UXP header, RTP timestamp, and marker bit will help the receiver to recover TB synchronization.

5.4 Framing and Timing Mechanism in UXP: The Info Stream

As described in Sect. 4, UXP creates its own packetization scheme by interleaving. The regular framing and timing structure of RTP is therefore destroyed. This section describes which kind of problems arise with interleaving and how they can be solved. This finally leads to the specification of the info stream.
The timestamp of an RTP packet usually describes the sampling time of the first octet included in the RTP data packet. This is in principle also true for UXP RTP packets. According to the time stamp definition in Sect. 5.1 every UXP RTP packet contains as timestamp the sampling time of the first octet in the corresponding TB. Therefore, all packets which belong to one TB contain the same RTP timestamp. This can lead to problems since due to the theoretical size limit of a TB (the limit for the number of columns is 256, and the limit for the number of rows is the maximum packet size), it can contain data from different sampling time instances, e.g. several video frames. Then the timing information of the later frames has to be determined from the media stream itself and not from the RTP timestamp.
A second problem arising with interleaving is that the framing mechanism of RTP is not supported. Since the payload of a single RTP-packet does not contain individually decodable payload, but rather the whole stream is reconstructed from a full TB, the UXP

RTP packets can not be used to provide information about the start of different access units within the octet stream.

The framing and time problem can be solved in many ways:

One solution of the problem would be to rely on the framing and timing mechanism of the elementary media stream. This is, for example, possible for media streams which contain start codes and information about the frame rate.
A second solution could be to define a specific framing mechanism for the info stream similar to [Laz04] and extend it by timing information. A third possibility is to insert the RTP packets of a media directly into a TB

In this specification, we consider only the first solution, i.e. to rely on the timing in the elementary media stream. Other solutions have to be defined as extensions of this specification. Therefore, an info-stream in this specification SHALL be defined as an elementary media stream which provides timing and framing.

5.5. In-band Signaling of the Structure of the Redundancy Profile

To enable a dynamic adaptation to varying link conditions, the actual redundancy profile used in the data TSB as well as the beginning and end of a TSB must be signaled to the receiving entity. Since out-of-band signaling either results in excessive additional control traffic, or prevents quick changes of the profile between successive TBs, an in-band signaling procedure is desired.
Since without knowledge of the correct redundancy profile, the decoding process cannot be applied to any of the erasure protection classes, the redundancy profile has to be protected at least as strongly as the most important element in the info stream. Therefore, an additional class EPC_P is used in the signaling TSB, where the number of parity symbols is by default set to the following value:
P=ceil(n/2.0)
Hence, up to 50% of the RTP packets can be lost, before the redundancy profile cannot be recovered anymore. This seems to be a reasonable value for the lowest point of operation over a lossy link. Alternatively, P may be explicitly signaled during session setup by means of SDP or H.245 protocol.
Consequently, since all other classes must have equal or less erasure protection capability, the maximum allowable value for class EPC_T in the data TSB is now limited to T<=P.
The signaling of the erasure protection vector is accomplished by means of descriptors. In the following we describe an efficient encoding scheme for the descriptors.
For each class EPC_i with R_i>0, there is a descriptor DP_i providing information about the size of class EPC_i (i.e. the

value of R_i) and establishing a relationship between the erasure protection of class EPC_i and that of the class EPC_(i+j), where j>0 and j is the smallest value for which R_(i+j)>0 is true. A descriptor DP_i is mapped onto one octet, which is sub-divided

into two half-octets (i.e. the higher and the lower four bits). The first half-octet is of type unsigned and contains the 4-bit representation of the decimal value R_i. The second half-octet is of type signed and contains the difference in erasure protection between class EPC_i and class EPC_(i+j), i.e. the signed 4-bit representation of the decimal value (-j) (where the MSB denotes the sign, and the lower three bits the absolute value). Note that the erasure protection P of class EPC_p is fixed, whereas the size R_P may vary.

Thus, the data to be filled into class EPC_P shall consist of a sequence of descriptors separated by stuffing indicators (see below), where the number of descriptors is primarily given by the number of protection classes EPC_i, 0<=i<=T, in the data TSB with R_i>0.

Without a-priori knowledge, the initial value for the size of the signaling TSB, R_P, should be set to one (row). When the number of necessary descriptors and stuffing indicators exceeds the (n-P) information positions, one or more additional rows have to be reserved. This is usually done by increasing the value for L_s to R_P>1, i.e. the data TSB is reduced to (L-R_P) rows. Hence, in order to indicate the actual size of the signaling TSB, an additional descriptor is inserted at the very beginning, which takes on the value 0xq0, where q denotes the (octal) four bit representation of the decimal value R_P.

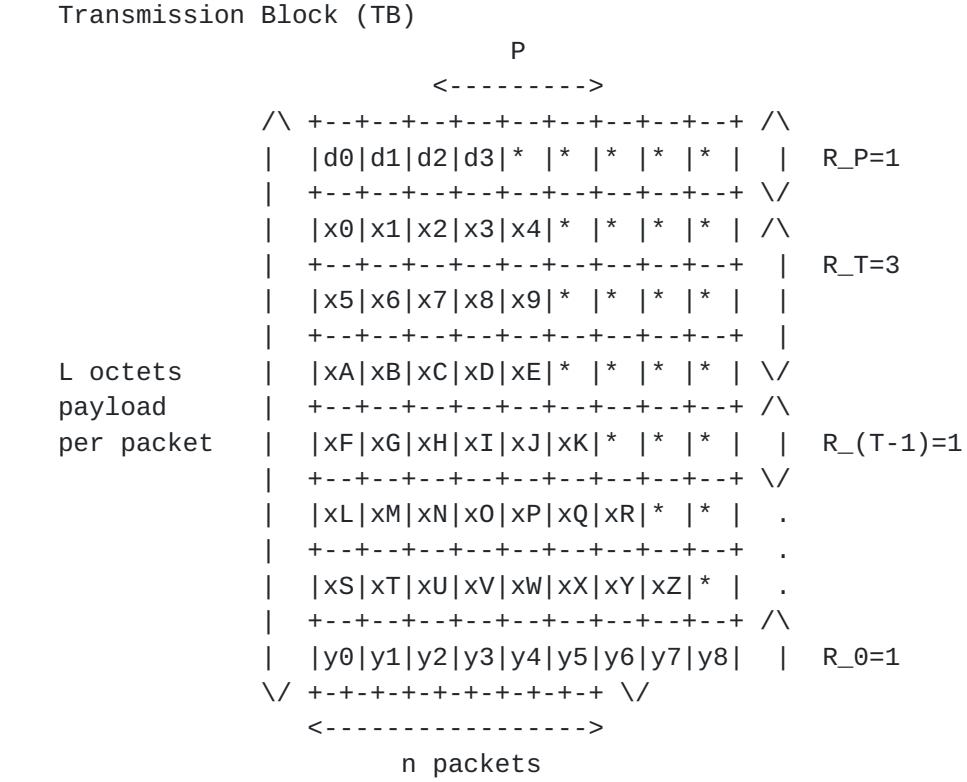
Furthermore, the end of each data TSB is signaled by the otherwise unused descriptor value 0x00, followed by exactly one stuffing indicator (SI). The latter is mapped onto an octet, which is of type unsigned and contains the 8-bit representation of the decimal value of the number of media stuffing symbols used at the end of the respective data TSB.

The (extended) sequence of descriptors and stuffing indicators is then mapped to the octet positions in the R_P rows of the signaling TSB from left to right and top to bottom. Each row is then encoded with the same (n,n-P) RS code.

If the number of descriptors and stuffing indicators is less than the available octet positions, however, empty positions in class EPC_P may be filled up with the otherwise unused descriptor 0x00.

At the receiving entity, the sequence of descriptors shall be recovered by performing erasure decoding on the first row of the TB (which definitely belongs to the signaling TSB) using the same algorithm as later for the data TSB. If successful, the very first descriptor now indicates the number of rows of the signaling TSB, and the next (R_P-1) rows are decoded to reconstruct the redundancy profile for the data TSB(s), together with the number of media stuffing symbols denoted by the respective SI(s).

The complete structure of the TB is now depicted in Fig. 6.



d# : descriptors and stuffing indicators for in-band signaling of the redundancy profile
x#,y# : info octets belonging to the info stream defined in Fig. 3
* : parity octets gained from Reed-Solomon coding

Fig. 6: General structure for UXP with in-band signaling of the redundancy profile

The following simple example is meant to illustrate the idea behind using descriptors: Let an erasure protection vector of length T+1=7 be given as follows:
EPV=(R_0,R_1,...,R_5,R_6)=(7,0,2,2,0,3,10)
Hence, the length L of the TB (including one row for the signaling TSB) is equal to 7+2+2+3+10+1=25 (rows/octets). If the width is assumed to be equal to 20 (columns/packets), then the erasure protection of the descriptors is P=10.
The corresponding sequence of descriptors can be written as
DP=(DP_6,DP_5,DP_3,DP_2,DP_0)=(0xAC,0x39,0x2A,0x29,0x7A),
where the values of the descriptors are given in hexadecimal notation. Next, the descriptor indicating the length of the

signaling TSB has to be inserted, the end of the data TSB has to be marked by 0x00, and the SI has to be appended. If the number of media stuffing symbols is assumed to be 3, the 10 info octets

in the signaling TSB take on the following values (descriptor stuffing included):
(0x10,0xAC,0x39,0x2A,0x29,0x7A,0x00,0x03,0x00,0x00)

5.6. Optional Concatenation of Transmission Sub Blocks

The following procedure may be applied if a single info stream would be too short to achieve an efficient mapping to a transmission block with respect to the fixed payload length L and the desired number of packets n. For example, intra-coded video frames (I-frames) are usually much larger than the following predicted ones (P-frames). In this case, a certain number z of successive small info streams should be each mapped to a transmission sub block with length L_d(y) and width n, such that L_d(1)+L_d(2)+...+L_d(z)=L_d.
The resulting transmission sub blocks can then be easily concatenated to form a TB of size L x n having one common signaling TSB (see Fig. 2): Since the second half-octet of the descriptors is of type signed (cf. Sect. 5.5.), we are able to signal both decreasing and increasing erasure protection profiles.
Again, we will give a simple example to illustrate this idea: Let the erasure protection vectors for two concatenated data TSBs be given as follows:
EPV1=(R1_0,R1_1,...,R1_5,R1_6)=(0,0,2,2,0,3,10),
EPV2=(R2_0,R2_1,...,R2_5,R2_6)=(0,0,2,2,0,3,10).
Hence, two single identical data TSBs will be concatenated to form a TB of length L=2*(2+2+3+10)+2=36 (rows/octets). If the width is again assumed to be equal to 20 (columns/packets), then the erasure protection of the descriptors is P=10. We reserve a total of two rows for the signaling TSB. The corresponding sequence of descriptors can now be written as DP=(0xAC,0x39,0x2A,0x29,0xA4,0x39,0x2A,0x29), where the values of the descriptors are given in hexadecimal notation. The values of the first four descriptors are taken from the descriptor of EPV1 as described in Sect. 5.5. (without the SI). The last four descriptors are taken from the descriptor of EPV2 (without SI) with one exception. The fifth descriptor of DP (i.e. 0xA4) is created as follows: The first half-octet is created according to Sect. 5.5. However, the second half-octet describes no longer the difference between R_P and R2_6. It rather describes the difference between R1_2 and R2_6, i.e. R1_2-R2_6, which can be a positive or negative number. If the number of media stuffing symbols is assumed to be 3 for each data TSB, the 20 info octet positions in the signaling TSB are filled with the following values (descriptor stuffing included):

```
   (0x20,0xAC,0x39,0x2A,0x29,0x00,0x03,0xA4,0x39,0x2A,0x29,0x00,0x03
   ,
   0x00,0x00,0x00,0x00,0x00,0x00,0x00)
```

Therefore from the example above, the following general rule MUST be used to create the resulting descriptors for concatenated data TSB #u and data TSB #v, where v=u+1:
Let EPVu=(Au_0,Au_1,...) and EPVv=(Av_0, Av_1,...) be the corresponding erasure protection vectors and DPu and DPv the corresponding descriptors created according to Sect. 5.5. (with stuffing). Let w be the smallest index for which Au_w >0. Let x be the largest index for which Av_x >0. The resulting descriptor can be created by concatenation of DPu and DPv where the first descriptor of DPv should be changed as follows:
The second half byte is defined by Au_w-Av_x.

6. Indication of UXP in SDP

From the discussion in Sect. 5.4 , we know that UXP encapsulates and protects the info stream. The info stream consists usually of a regular RTP-Payload format, e.g. RFC 3016.
There is no static payload type assignment for UXP, so dynamic payload type numbers MUST be used. The binding to the number is indicated by an rtpmap attribute. The name used in this binding is
"UXP". The payload type number of UXP is indicated in the "m" line of the
media, as well as the payload type of the info-stream.

A sample indication of UXP in SDP is as follows:

```
m = video 8000 RTP/AVP 98 99
a = rtpmap:98 UXP/90000
a = rtpmap:99 MP4V-ES/90000
```

Here, PT 98 indicates that the payload consists of UXP with the corresponding info stream "MP4V-ES". Alternatively, PT 99 can be used which indicates "MP4V-ES" without UXP.
Since UXP is generic, several payload types can be protected. The lines

```
m = video 8000 RTP/AVP 98 99 100
a = rtpmap:98 UXP/90000
a = rtpmap:99 MP4V-ES/90000
a = rtpmap:100 H263-1998/90000
```

mean that UXP can be used with either "MP4V-ES" or "H263-1998" as

info stream (indicated by PT 98 in the RTP-Header and either block PT=99 or block PT=100 in the UXP-Header). Alternatively,

PT=99 or PT=100 in the RTP-Header means the use of "MP4V-ES" or "H263-1998" without UXP.

As described in Sect. 5.5., the parameter P has the default value P=ceil(n/2.0), if not otherwise stated. The parameter P MAY be specified explicitly by means of SDP:

a = fmtp:98 UXP-prof: fvalue

where fvalue is a floating point number in the interval (0 < fvalue <1) and specifies P by P=ceil(n*fvalue). For example, if we set fvalue=0.5,

a = fmtp:98 UXP-prof: 0.5

we get the default value for P, since P=ceil(n/2.0).
The ABNF for fvalue according to RFC 2234 is

fvalue = "0" "." 1*2DIGIT

7. Security Considerations

The payload of the RTP-packets consists of an interleaved media and parity stream. Therefore, it is reasonable to encrypt the resulting stream with one key rather than using different keys for media and parity data. It should also be noted that encryption of the media data without encryption of the parity data could enable known-plaintext attacks.
The overall proportion between parity octets and info octets should be chosen carefully if the packet loss is due to network congestion. If the proportion of parity octets per TB is increased in this case, it could lead to increasing network congestion. Therefore, the proportion between parity octets and info octets per TB MUST NOT be increased as packet loss increases due to network congestion.
The overall transmission rate for parity and info octets MUST be controlled by a congestion control algorithm. The congestion control algorithm used for the media which is protected by UXP MUST by used for the overall transmission rate for parity and info octets in UXP, i.e. for the resulting data rate. The trade-off between parity and info octets is determined by the optimization algorithm which determines the EPV and is, thus, out of scope of this specification.

8. IANA Considerations

8.1 Video

To: ietf-types@iana.org

Subject: Registration of MIME media type video/UXP

MIME media type name: video

MIME subtype name: UXP

Required parameters: none

[RFC3555] mandates that RTP payload formats without a defined rate must define a rate parameter as part of their MIME registration. This payload specification does not specify a rate parameter. However, the rate for UXP payload is equal to the rate of the media data it protects.

Optional parameters:
UXP-prof: Describes the redundancy of the signaling sub block (cf. Sec.5.5.).

Encoding considerations: This format is only defined for transport within the Real Time Transport protocol (RTP) [RFC3550]. Its transport within RTP is fully specified within this specification.

Security considerations: The same security considerations apply to these mime registrations as to the payloads for them, as detailed in this specification.

Interoperability considerations: none

Published specification: This MIME type is described fully within this specification.

Applications which use this media type: Audio and video streaming tools which seek to improve resiliency to loss by sending additional data with the media stream.

Additional information: none

Person & email address to contact for further information:

Marcel Wagner
Siemens AG
Otto-Hahn-Ring 6

81730 Munich, Germany
email: Marcel.Wagner@siemens.com

Intended usage: COMMON

Author/Change controller: Marcel Wagner.

RTP and SDP Issues: Usage of this format within RTP and the Session Description Protocol (SDP) [RFC2327] are fully specified within this specification.

8.2 Audio

To: ietf-types@iana.org

Subject: Registration of MIME media type audio/UXP

MIME media type name: audio

MIME subtype name: UXP

Required parameters: none

[RFC3555] mandates that RTP payload formats without a defined rate must define a rate parameter as part of their MIME registration. This payload specification does not specify a rate parameter. However, the rate for UXP payload is equal to the rate of the media data it protects.

Optional parameters:
UXP-prof: Describes the redundancy of the signaling sub block (cf. Sec.5.5.).

Encoding considerations: This format is only defined for transport within the Real Time Transport protocol (RTP) [RFC3550]. Its transport within RTP is fully specified within this specification.

Security considerations: The same security considerations apply to these mime registrations as to the payloads for them, as detailed in this specification.

Interoperability considerations: none

Published specification: This MIME type is described fully within this specification.

Applications which use this media type: Audio and video streaming tools which seek to improve resiliency to loss by sending additional data with the media stream.

Additional information: none

Person & email address to contact for further information:

Marcel Wagner
Siemens AG
Otto-Hahn-Ring 6
81730 Munich, Germany
email: Marcel.Wagner@siemens.com

Intended usage: COMMON

Author/Change controller: Marcel Wagner.

RTP and SDP Issues: Usage of this format within RTP and the Session Description Protocol (SDP) [RFC2327] are fully specified within this specification.

9. Application Statement

There are currently two different schemes proposed for unequal error protection in the IETF-AVT: Unequal Level Protection (ULP) and Unequal Erasure Protection (UXP).
Although both methods seem to address the same problem, the proposed solutions differ in many respects. This section tries to describe possible application scenarios and to show the strengths and weaknesses of both approaches.
The main difference between both approaches is that while ULP preserves the structure of the packets which have to be protected and provides the redundancy in extra packets, UXP interleaves the info stream which has to be protected, inserts the redundancy information, and thus creates a totally new packet structure.
Another difference concerns multicast compatibility: It cannot be assumed that all future terminals will be able to apply UXP/ULP. Therefore, backward compatibility could be an issue in some cases. Since ULP does not change the original packet structure, but only adds some extra packets, it is possible for terminals which do not
support ULP to discard the extra packets. In case of UXP, however, two separate streams with and without erasure protection have to be sent, which increases the overall data rate.
Next, both approaches offer different mechanisms to adjust packet sizes, if necessary: UXP allows to adjust the packet sizes arbitrarily. This is an advantage in case the loss probability is dependent on the packet length, which happens, for example, if the end-to-end connection contains wireless links. In this case proper adjustment of the packet size is one essential network adaptation technique. In addition, if a preencoded stream is sent over the network, the packet size can be adjusted independently

of slice structures.
Since ULP does not change the existing packetization scheme, this flexibility does not exist.

The ability of UXP to adjust the packet size arbitrarily can be especially exploited in a streaming scenario, if a delay of several hundred milliseconds is acceptable. It is then possible to fill several video frames into a single TB of desired size, e.g. a group of pictures consisting of I-frame, P-frames and B-frames. The redundancy scheme can thus be selected in such a way as to guarantee the following property: In case of packet loss, the P-frames are only recoverable if the I-frame on which the decoding of P-frames depends is recoverable. The same is true for B-frames, which can only be decoded if the respective P-frames are recoverable. This prevents situations in which, for example, the B-frames have been received correctly, but the P-frames have been lost, i.e. assures a gradual decrease in application quality also on the frame level. Of course, a similar encoding is possible with ULP. But in this case one might have to send several frames within one packet which leads to large packet sizes.
Furthermore, decoding delay is also a crucial issue in communications. Again, both approaches have different delay properties: UXP introduces a decoding delay because a reasonable amount of correctly received packets are necessary to start decoding of a TB. The delay in general depends on the dimensions of the interleaver. This should be considered for any system design which includes UXP.
With ULP, every correctly received media packet can be decoded right away. However, a significant delay is introduced, if packets are corrupted, because in this case one has to wait for several redundancy packets. Thus, the delay is in general dependent on the actual ULP-FEC-packet scheme and cannot be considered in advance during the system design phase.
Finally, we want to point out that UXP uses RS codes which are known
to be the most efficient type of block codes in terms of erasure correction capability.

10. Intellectual Property Considerations

Siemens AG has filed patent applications that might possibly have technical relations to this contribution.
On IPR related issues, Siemens AG refers to the Siemens Statement on Patent Licensing, see http://www.ietf.org/ietf/IPR/SIEMENS-General.

The following patent might apply to this specification:
United States Patent 5,617,541, April 1, 1997, System for packetizing data encoded corresponding to priority levels where reconstructed data corresponds to fractionalized priority level

and received fractionalized packets, Inventors: Albanese; Andres (Berkeley, CA); Luby; Michael G. (Berkeley,CA); Bloemer; Johannes F. (Berkeley, CA); Edmonds; Jeffrey A. (Berkeley, CA) Filed: December 21, 1994

11. References

Normative References

[RFC2733] J. Rosenberg and H. Schulzrinne, "An RTP Payload Format for Generic Forward Error Correction", Request for Comments 2733, Internet Engineering Task Force, Dec. 1999.

[Lin83] Shu Lin and Daniel J. Costello, Error Control Coding: Fundamentals and Applications, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1983.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: A Transport Protocol for Real-time Applications", RFC 3550, July 2003.

[RFC3555] Casner, S., Hoschka, P.," MIME Type Registration of RTP Payload Formats", RFC 3555, July 2003

[RFC2327] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

Informative References

[Alb96] A. Albanese, J. Bloemer, J. Edmonds, M. Luby, and M. Sudan, "Priority encoding transmission", IEEE Trans. Inform. Theory, vol. 42, no. 6, pp. 1737-1744, Nov. 1996.

[Li01] W. Li: "Streaming video profile in MPEG-4", IEEE Trans. on Circuits and Systems for Video Technology, Vol. 11, no. 3, 301-317, March 2001.

[Bla00] G. Blaettermann, G. Heising, and D. Marpe: "A Quality Scalable Mode for H.26L", ITU-T SG16, Q.15, Q15-J24, Osaka, May 2000.

[Bur99] F. Burkert, T. Stockhammer, and J. Pandel, "Progressive A/V coding for lossy packet networks - a principle approach", Tech. Rep., ITU-T SG16, Q.15, Q15-I36, Red Bank, N.J., Oct. 1999.

[Lie99] Guenther Liebl, "Modeling, theoretical analysis, and coding for wireless packet erasure channels", Diploma Thesis, Inst. for Communications Engineering, Munich University of Technology, 1999.

[Hor99] U. Horn, K. Stuhlmuller, M. Link, and B. Girod, "Robust Internet video transmission based on scalable coding and unequal error protection", Image Com., vol. 15, no. 1-2, pp. 77-94, Sep. 1999.

[Wen02] S. Wenger, "H.26L over IP: The IP-Network Adaptation Layer", Packet Video 2002, Pittsburgh, Pennsylvania, USA, April 24-26,2002.

[Laz04]Lazzaro, John, "Framing RTP and RTCP Packets over Connection-Oriented Transport", draft-ietf-avt-rtp-framing-contrans-02.txt, work in progress, 2004

12. Acknowledgments

Many thanks to Magnus Westerlund, Philippe Gentric, Stephen Casner, and Hermann Hellwagner for helpful comments and

improvements. The authors would like to thank Thomas Stockhammer who came up with the original idea of UXP. Also, the help of Gero Baese, Frank Burkert, and Minh Ha Nguyen for the development of UXP is well acknowledged.

13. Author's Addresses

Guenther Liebl
Institute for Communications Engineering (LNT)
Munich University of Technology (TUM)
D-80290 Munich
Germany
Email: {liebl}@lnt.e-technik.tu-muenchen.de

Marcel Wagner
Siemens AG - Corporate Technology CT IC 2
D-81730 Munich
Germany
Email: marcel.wagner@siemens.com

Juergen Pandel
Siemens AG - Corporate Technology CT IC 2
D-81730 Munich
Germany
Email: juergen.pandel@siemens.com

Wenrong Weng
Siemens AG - Corporate Technology CT IC 2
D-81730 Munich
Germany
Email: wenrong.weng@siemens.com

14. Full Copyright Statement

15. Intellectual Property Notice

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.