Remote Collaborative Real-Time Multimedia Experience over the Future Internet

ROMEO

Grant Agreement Number: 287896

D4.2

Report on streaming/broadcast techniques for 3D multi-view video and spatial audio

Document description	
Name of document	Report on streaming/broadcast techniques for 3D multi-view video and spatial audio
Abstract	This document provides a detailed description of the packetization schemes in ROMEO and specifies high level syntax elements of the media formats in order to perform efficient transport and synchronization of the 3D audio and multiview video streams. Adaptation mechanisms and error concealment methods are also proposed in the context of degraded network conditions.
Document identifier	D4.2
Document class	Deliverable
Version	1.0
Author(s)	N.Tizon, D. Nicholson (VITEC) H. Weigold, H. Ibl, J. Lauterjung (R&S) K. Birkos, A. Kordelas, A. Lykourgiotis, I. Politis (UPAT) Xiyu Shi (MulSys) M.Laabs (IRT) E. Ekmekcioglu (UNIS) A. Akman, S. O. Pelvan, S. Çiftçi, E. Çimen Öztürk (TTA)
QAT team	D. Doyen (TEC) F. Pascual Blanco (TID) H. Marques (IT)
Date of creation	24-Jul-2012
Date of last modification	21-Dec-2012
Status	Final
Destination	European Commission
WP number	WP4
Dissemination Level	Public
Deliverable Nature	Report

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Purpose of the Document

This document (D4.2 Report on streaming/broadcast techniques for 3D multi-view video and spatial audio) aims at specifying the interface and the wrapping mechanisms between the media application layer and the network transport layer. These specifications rely on existing standard protocol and encapsulation approaches and also consist in the definitions of new packetization schemes and packet processing in the network.

1.2 Objectives and Achievements

In the framework of the ROMEO project and especially in the WP4, this document will serve as a technical specification for the development of all the modules that are involved in the transport/streaming of the 3D ROMEO content:

- Low network layers: the P2P protocol implementation in ROMEO must be done according to the specific ROMEO application constraints such as the nature of the targeted networks (physical layer), the required level of interactivity and the kind of 3D audiovisual content.
- In terms of reliability and error correction, based on the application class targeted in ROMEO, each layer will deliver a service with a given level of QoS.
- In addition to the classical OSI layering approach, network adaptation mechanisms based on cross-layering methods will be implemented in order to optimize the transport in heterogeneous environments.

1.3 Structure of the Document

This deliverable is structured as follows:

- Section 2 describes the low layer requirements and specifications for packetization
- Section 3 describes the proposed packetization scheme
- Section 4 deals with network adaptation and error resilience

2 Low layer requirements and specification for packetization

2.1 MPEG-TS requirements

2.1.1 General principle

The MPEG-2 Transport Stream format specification [7] (a.k.a MPEG-2 TS, or TS) describes how to combine one or more elementary streams of video and audio, as well as other data, into single or multiple streams which are suitable for storage or transmission as illustrated in Figure 1. MPEG Transport Stream is widely used, and is the basis for digital television services and applications.

Moreover, this document provides additional information to enable synchronized decoding and buffer management.

In the scope of the ROMEO project, only the Transport Stream specifications will be considered. A Transport Stream can be a combination of several programs (e.g. TV channels). In the framework of the ROMEO project, the Transport Stream will combine only one program with one time base. A program is made of elementary streams (audio, video and/or metadata) which share this common time base. A transport stream is a continuous succession of fixed size and mixed (can be video packets, audio packets, metadata and other data packets) packets.

MPEG-2 Transport Stream can contain several type of Elementary Streams such as MPEG-2 Video (ISO/IEC 13818-2), MPEG-4 AVC/SVC and MVC (ISO/IEC 14496-10), MPEG Audio (ISO/IEC 11172-3, ISO/IEC 13818-3…), Dolby Digital AC3, E-AC3 and others as well as metadata (e.g. MPEG-7 and SMPTE 336) and other types of information such as Electronic Program Guides (EPG), including proprietary information thanks to the user private data stream type.

Each of this elementary streams type is defined in MPEG-2 TS standard and the addition of a new supported format requires an amendment to ISO/IEC 13818-1, as it is currently done for the new HEVC video coding standard.

The ROMEO elementary streams to be multiplexed will be the following:

- MPEG-4 AVC/SVC
- Depth map
- Metadata
- Audio

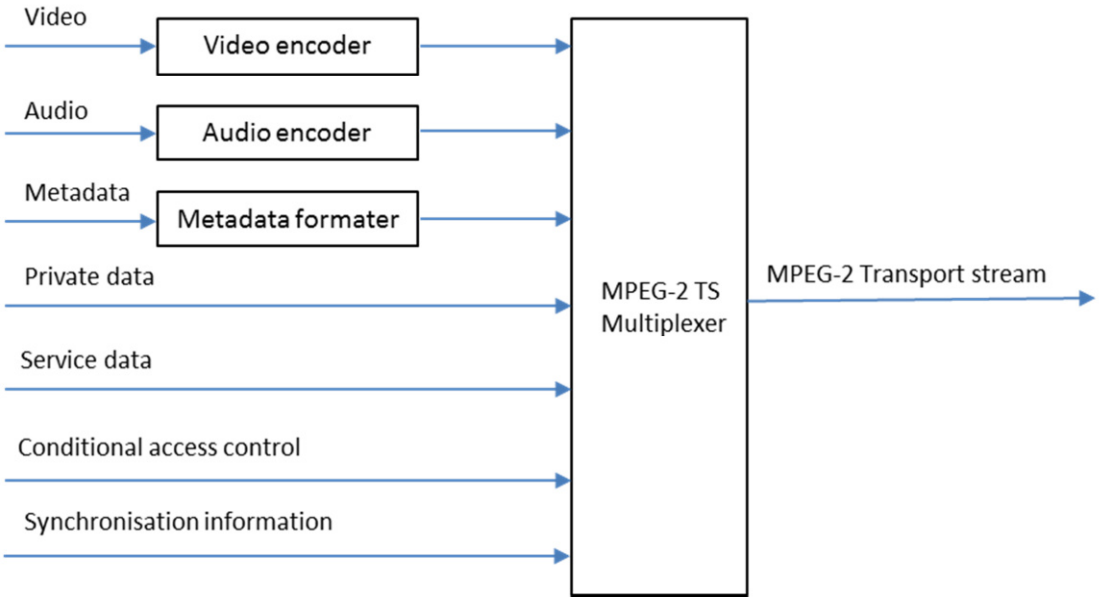

Figure 1 - General principle of a Transport Stream multiplexer

A TS packet starts with a sync byte and a header (see Figure 2). Additional optional transport fields, as signalled in the optional adaptation field, may follow. The rest of the packet consists of payload. Packets are 188 bytes in length among which 4 bytes are dedicated to the header. It has to be noted that for recording applications (such as blu-ray disks) an optional 4 bytes header can take place before the Sync Byte, carrying a time stamp information, useful for fast random access. This option is not part of the MPEG-2 TS standard but is now commonly used.

The first byte of the TS packet header is a synchronization byte that has the value 47 (hex). Transport error indicator bit determines if the TS packets contains remaining errors after error correction process (e.g. Reed-Solomon FEC in DVB-S). Payload Unit Start indicator signals that the first byte of the TS payload is also the first byte of a Packetized Elementary Stream, which includes an Elementary Stream. The Continuity Counter is incremented along successive TS packets belonging to the same Packetized Elementary Stream and is used for being able to detect packet losses.

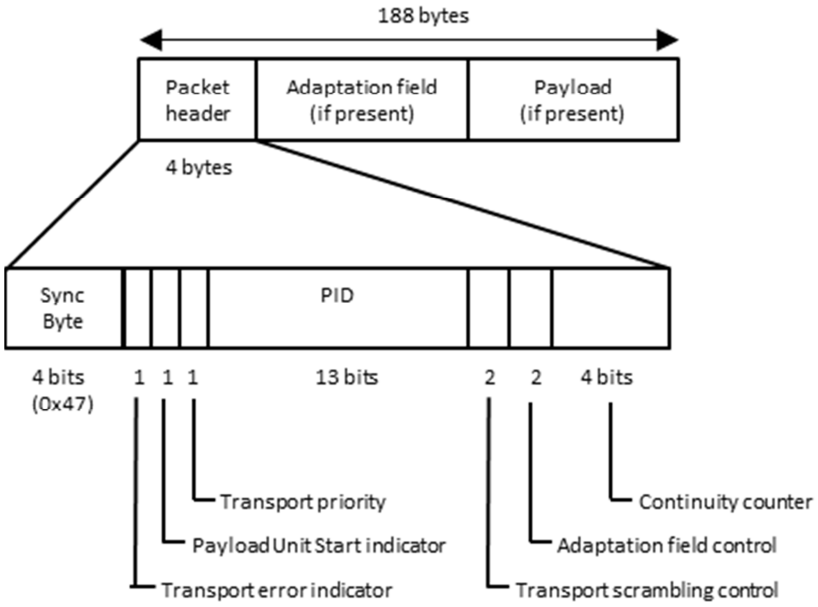

Figure 2 - MPEG2-TS packet header

A 13-bit packet ID (PID) identifies each table or elementary stream in a transport stream. A demultiplexer extracts elementary streams from the transport stream in part by looking for packets identified by the same PID. Apart from 17 reserved values for specific purposes, the PID can take one of the 8175 available values, which corresponds therefore to the maximum number of elementary streams present in a transport stream.

Within the 17 reserved values, it exists a TS packet with a unique PID value of 0x0000 called Program Association Table (PAT) that lists the PIDs of the tables called Program Map Table (PMT) describing each program.

Each single program is described by a PMT in which the PIDs of the TS packets associated with that program are listed. PAT and PMT relationship in MPEG2-TS is depicted in Figure 3.

A program is made of video and audio streams which are packetized in the form of Packetized Elementary Streams (PES) packets.

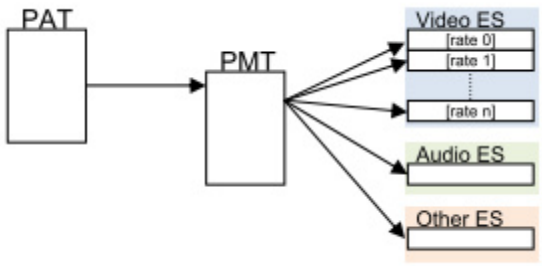

Figure 3 - PAT and PMT relationship in MPEG2-TS

When the data transmission scheme imposes strict constant bitrate requirements on the transport stream, some additional packets that containing no data can be inserted by the multiplexer. PID 0x1FFF is reserved for this purpose.

To enable a decoder presenting synchronized audio and video content, a Program Clock Reference (PCR) is transmitted in the adaptation field of an MPEG-2 transport stream packet. The PCR is associated to a unique PID and is identified by the pcr_pid value in the corresponding Program Map Table.

Transport Streams are logically constructed from PES (Packetized Elementary Stream) packets (see Figure 4). The Transport Stream system layer is divided into two sub-layers, one for multiplex-wide operations (the Transport Stream packet layer), and another for stream-specific operations (the PES packet layer). The PES Stream is a logical construction and is not defined as a stream for interchange and interoperability purposes. For Transport Streams, the packet length is fixed while both fixed and variable PES packet lengths, which may be larger than TS packet length, are allowed. PES packets can be of variable length (with a maximum length of 64kB). A PES packet consists of a header and a payload, which are data bytes of the Elementary Stream. There are no requirements to align the Access Units (i.e. the video frames in the case of a video elementary stream) to the start of a PES packet payload. A new access unit can start at any point in the payload of a PES packet and several access units can be contained in a single PES packet.

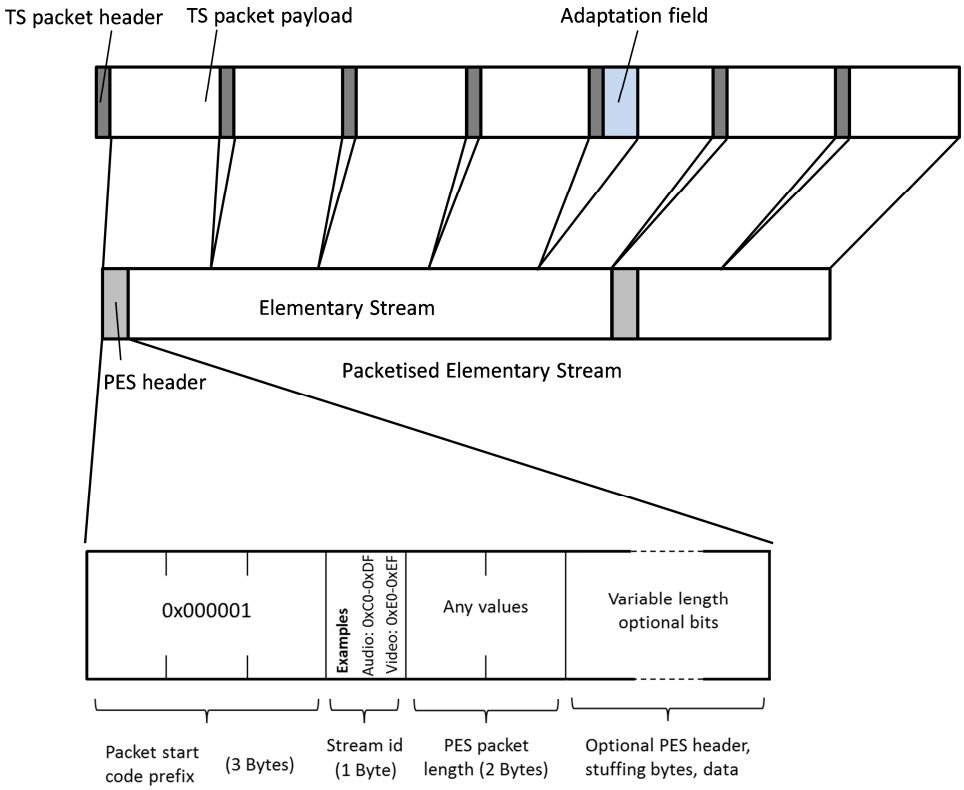

Figure 4 - PES packet

A PES packet header starts with a prefix code followed by the stream_id that allows distinguishing PES packets that belong to the same Elementary Stream or to another

Elementary Stream of the same program. For SVC video streams all video sub-bitstreams of the same video stream shall have the same stream_id value.

Following bits includes the PES packet length and two flags indicating the presence of optional information such as copyright information or times stamps (PTS/DTS). Time stamps are the mechanism to ensure correct synchronization between the several Elementary Streams of the same program.

The value of the Program Clock Reference (PCR) is employed to generate a System Time Clock (STC) in the decoder. The STC provides a highly accurate time base that is used to synchronize audio and video elementary streams. This synchronization is using the PTS/DTS (Presentation Time Stamp/Decoding Time Stamp) information. Clocks used at the multiplexer and decoder are measured in units of 27MHz coded in 42 bits. Each program of a multiplex has its own independent clock which needs not to be synchronized with clocks of other programs of the multiplex. PCR must appear at least every 0.1s. The Presentation Time Stamp specifies the time in which the decoded access unit (the content) has to be presented to the viewer. The Decoding Time Stamp specifies the time an access unit has to be decoded. DTS is necessary when B pictures are present in the video Elementary Stream as their order of arrival at the input of the decoder is not in the presentation order. A DTS is always accompanied with a PTS. PTS will be equal or greater than its associated DTS. For most types of Elementary Streams, PTS will be the only time stamps to be needed. Time Stamps are expressed in units of 90kHz and coded in 33 bits. In order to ease the synchronization process to be developed in ROMEO and to make it more robust toward packet losses, as the requested bandwidth of PTS and DTS information is quite low, PTS and DTS information will be added to all PES packets.

2.1.2 Multiplex of MPEG-4 SVC Elementary Streams

SVC is part of MPEG-4 AVC/H.264 specification and detailed in Annex G of ISO/IEC 14496-10 | ITU-T H.264. SVC provides scalability above an AVC bitsream which is used as a base layer and can be decoded independently from the enhancement layer(s), in particular in the case the decoder is only AVC capable.

In ROMEO, for compliance and interoperability, the Scalable High Profile level 4 will be supported.

As for AVC, SVC video data is included into NAL (Network Abstraction Layer) units (NALUs) called VCL NAL, each of them being a packet that contains an integer number of bytes. The first byte of each NAL unit is a header byte that contains an indication of the type of data in the NAL unit, and the remaining bytes contain payload data of the type indicated by the header.

NAL units contains video data (VCL) or other information types, which is associated to additional information such as sequence parameter set (SPS), and picture parameter set (PPS), and optional supplemental enhancement information (SEI). NAL Units of type 0 contains SPS. The SPS contains important information necessary for the decoding of the video sequence, whereas PPS contains important information that is necessary for the decoding of one or more pictures in the sequence.

NAL Units of type 8 contains PPS and NAL Units of type 6 contains SEI. SEI is described in detail in Annex E of ISO/IEC 14496-10. SVC requires the addition of SPS extension and PPS for the extension slices.

A video sequence consists of a sequence of access units with only one SPS and can be decoded independently from any other coded video sequence. Each encoded picture belongs to one single access unit that can be signalled by the use of an Access unit delimiter NAL

(NAL type 9). VCL NALUs of an access unit consists in a set of slice data belonging to the same picture. At the beginning of a coded video sequence, the first picture is included in an IDR (Instantaneous Decoding Refresh) access unit. An IDR access unit contains an intra-picture, decodable indepently from the other pictures, that can be followed by either other IDR access units or non IDR (N-IDR) access units and contains pictures encoded by using prediction mechanism.

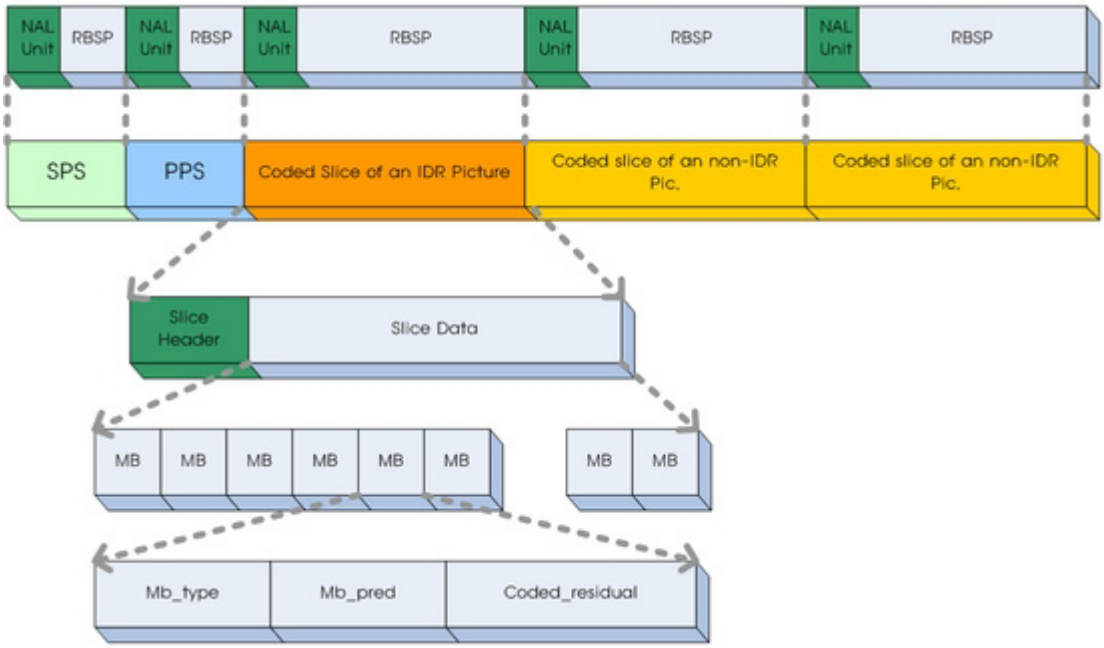

Figure 5 - MPEG-4 AVC bitstream structure

Each NAL unit has one or three byte header depending on the type of its NAL unit payload of variable byte length, which is called Raw Byte Sequence Payload (RBSP). The overall MPEG-4 AVC bitstream structure is depicted in Figure 5.

SVC NALs that carries parameter data or RBSP of spatial base layer (H.264/MPEG AVC compatible) needs only one byte header to describe the NAL type. This NAL header consists in one forbidden bit (F), two bits indicating if the NAL unit is used for prediction and five bits to indicate the NAL unit type as depicted in Figure 6. At the end of the payload, trailing bits are used to adjust the payload size to become a multiple of bytes.

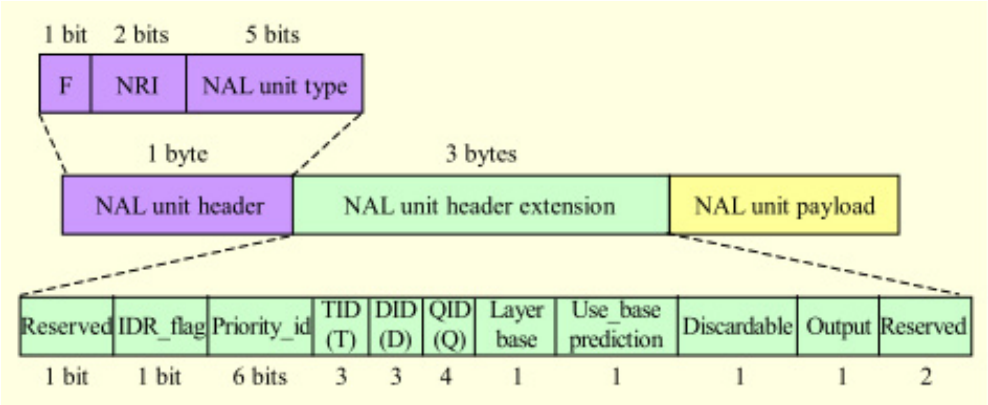

Figure 6 - SVC NALU structure

SVC enhancement layer video data is included in NAL Units of type 20 which allows differentiating it from AVC base layer NAL units. They need three bytes for their header to describe scalability related information such as spatial, temporal, quality level, priority of the data in the NAL as illustrated in Figure 7, where:

- R is a reserved bit that if set to 1 signal the presence of SVC NAL unit header extension and payload, otherwise signals the presence of a MVC NAL unit header extension and payload,
- I is idr_flag, equal to 1 specifies that the current coded picture is an IDR picture otherwise equal to 0,
- N is no_inter_layer_pred_flag specifies whether inter-layer prediction may be used for decoding the coded slice,
- Priority_id specifies a priority identifier for the NAL unit,
- Dependency_id specifies a dependency identifier for the NAL unit,
- Quality_id specifies a quality identifier for the NAL unit,
- Temporal_id specifies a temporal identifier for the NAL unit,
- U is use_ref_base_pic_flag, equal to 1 specifies that reference base pictures (when present) and decoded pictures (when reference base pictures are not present) are used as reference pictures for inter prediction,
- D is discardable_flag equal to 1 specifies that the current NAL unit is not used in the decoding process of NAL units of the current coded picture and all subsequent coded pictures that have a greater value of dependency_id than the current NAL unit.
- O is output_flag,
- RR are reserved 2bits, shall be equal to 3.

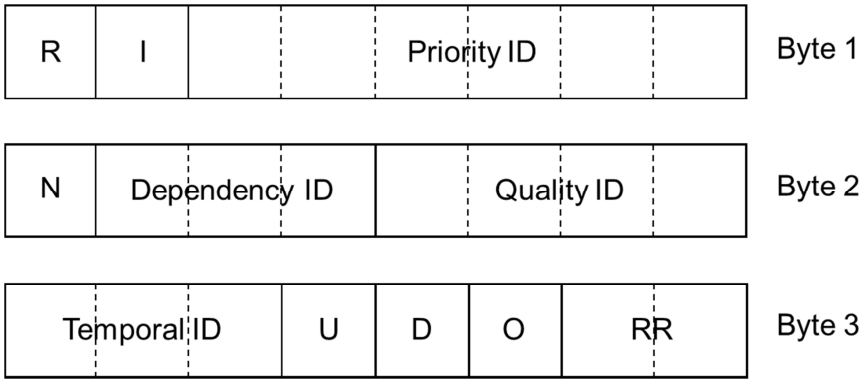

Figure 7 - SVC NALU header extension

The SVC base layer, i.e. the AVC video sub-bitstream has a dependency_id equal to 0.

It has to be noted that MVC data (Multiview coding, described in Annex H of ISO/IEC 14406-10 | Rec ITU-T H.264) is also using NAL Type 20 for its NALUs, which provides interoperability between SVC and MVC extensions to AVC. In this case the specific SVC NAL extension header is replaced by a MVC NAL extension header.

Aspect ratio, timing information, picture colorimetry and picture chrominance locations, and picture structure information of an SVC system are signalled by using Video Usability Information (VUI) parameters inside the Sequence Parameter Set.

Even if SEI is optional for being able to decode the video information but it provides useful information, that can be mandatory for some applications like Blu-Ray BD3D and 3D on smartphones which uses side by side representation. All SEI messages that apply to SVC enhancement layers should be included in the AVC video base layer.

The use of MPEG-2 TS as a transport layer for AVC/SVC requires NALUs as an ordered stream of bits within which the locations of NAL unit boundaries need to be identifiable by a patternto prevent emulation of the corresponding pattern within the compressed data.

AVC/H.264 specification defines a byte stream format (Annex B of ISO/IEC 14496-10 | Rec ITU-T H.264) in which, each NAL unit is prefixed by a synchronization byte sequence (0×000001 or 0×00000001). This byte sequence is called a start code prefix (see Figure 8), is inserted before every NAL Unit, to create a new H.264 elementary stream ready to be multiplexed into a MPEG-2 Transport Stream.

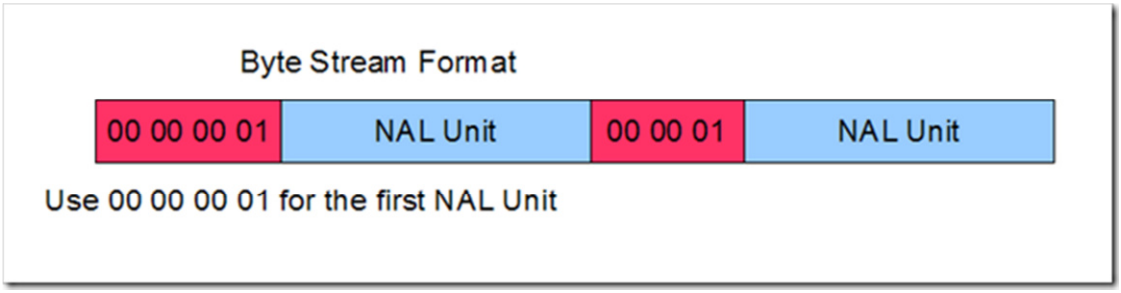

Figure 8 - MPEG-4 AVC/H.264 Byte Stream format

The boundaries of the NAL unit are then identified by searching the coded data for the unique start code prefix pattern. The use of emulation prevention bytes guarantees that start code prefixes are unique identifiers of the start of a new NAL unit. The different NAL units related to video streams in ROMEO are listed in Table 1. It has to be noted that for transport of AVC/SVC directly over RTP (RFC6184, RFC 6190), NAL units are carried in data packets without start code prefixes.

Content of NAL unit	**nal_unit_type**
Coded slice of a non-IDR picture	1
Coded slice of an IDR picture	5
Supplemental enhancement information (SEI)	6
Sequence parameter set (SPS)	7
Picture parameter set (PPS)	8
Access unit delimiter	9
Sequence parameter set extension	13
Prefix NAL unit	14
Coded slice extension (SVC or MVC VCL data)	20

Table 1 - ROMEO related NAL unit types

2.1.3 Audio Elementary Streams

The audio Elementary Stream for the ROMEO project uses the Audio Data Transport Stream (ADTS) structure as specified in ISO/IEC 13818-7. In the ADTS structure, each encoded raw audio frame is preceded by a header that contains data necessary for the decoding. This configuration is useful for streaming applications because the decoding information can easily be retrieved.

An ADTS frame consists of a fixed header, a variable header, an optional error check, and a specified number of raw data blocks. The fixed header consists of general information for streaming, such as the sampling rate, channels, and profile, which remains the same in every frame. The variable header on the other hand contains frame-related information such as variable frame size. The header size without or with CRC is 7 or 9 bytes.

The ADTS frame structure is illustrated in Figure 9 and the fields in the header are described in Table 2.

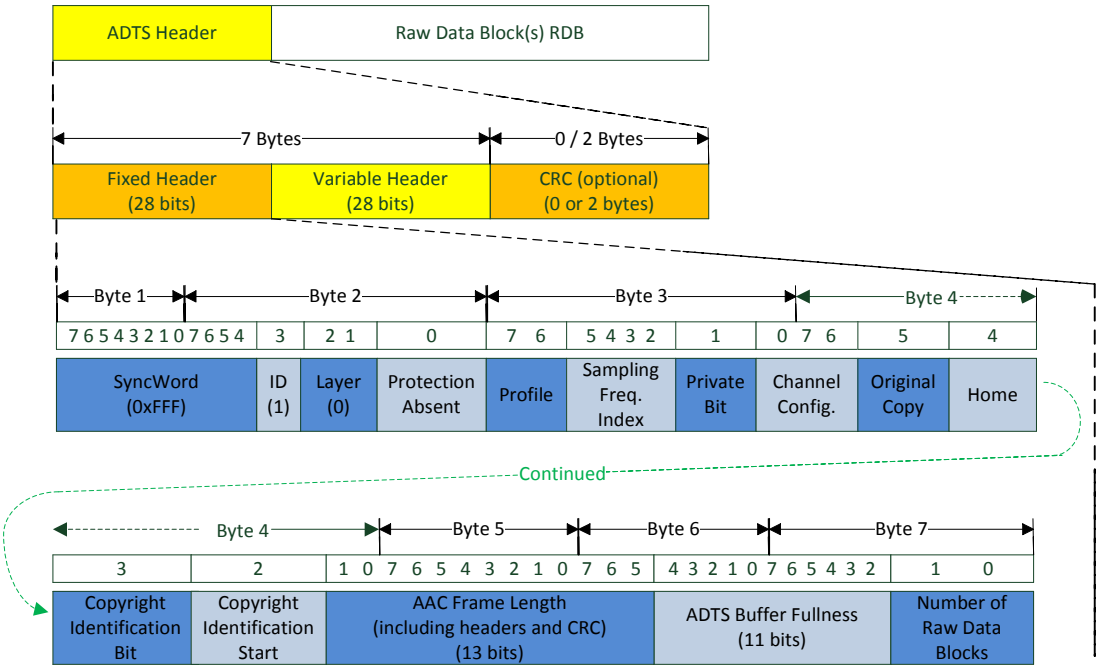

Figure 9 - MPEG-4 ADTS frame structure

Field name	No. of bits	Remarks
Syncword	12	All bits must be 1 (0xFFF)
ID	1	MPEG identifier, set to 1
Layer	2	Indicates which layer is used. Always 00
Protection-absent	1	Indicates whether error checking field CRC is present or not. 1 if no CRC / 0 if CRC exists
Profile	2	Profile used, 0 for Main profile, 1 for Low Complexity (LC) profile
Sampling frequency index	4	Indicates the sampling frequency used. 3 for 48KHz

Private bit	1	Private bit, only informative. Set to 0 when encoding, ignore when decoding
Channel configuration	3	Indicates channel configuration used. 6 for surround 5.1 configuration
Original copy	1	Indicates originality. Set to 0 when encoding, ignore when decoding
Home	1	Original or copy. Set to 0 when encoding, ignore when decoding
Copyright identification bit	1	One bit of the 72-bit copyright identification field. Set to 0 when encoding, ignore when decoding.
Copyright identification start	1	One bit to indicate the first bit of copyright identification. Set to 0 when encoding, ignore when decoding.
AAC Frame length	13	Frame length including 7- or 9-byte header length
ADTS buffer fullness	11	State of buffer fullness, 0x7FF means variable rate bit stream
Number of raw data blocks in the ADTS frame	2	No. of raw data blocks (RDB) in ADTS frame minus 1. Minimum value is 0 indicating 1 RDB.
CRC check	16	Optional. 2 bytes CRC if protection-absent is 0. Otherwise not present.

Table 2 - Description of MPEG-4 ADTS frame header

2.2 P2P network/protocol requirements

ROMEO chunk header will include the fields indicated in Table 3. The chunk header has a size of 14 bytes and includes the content related information such as priority and layer identification which allows the ROMEO P2P system to have an adaptive, content aware delivery mechanism. With this header structure, there will be no need to extract information for the peer components during chunk forwarding and chunks selection.

Header Parameter	**Purpose**	**Size (in bits)**
Content Identifier	Unique id for the content to match with DVB stream	4
Chunk identifier	Unique id, used for reporting defect P2P chunks	16
PID (Stream id)	Packet ID, used for identifying different elementary streams	13
PCR	Program clock reference, used for synchronising P2P and DVB-T2 streams	33
Metadata flag	If set, the payload contains metadata (overrides audio flag)	1

View identifier	Identifies different video views	3
Descriptor number	Indicates the different descriptors	3
Layer flag	Indicates the layer type: either base or enhancement layer	3
Audio flag	If set, the payload contains audio data	1
Payload size	Size of the payload (number of bytes)	16
CRC	CRC value for the payload	16
Priority	Indicates priority of the stream to decide if the stream is discardable	3

Table 3 - P2P Header fields

2.3 DVB network/protocol

The DVB-T2 system is described in the ETSI standard EN 302 755 [8]. As shown in Figure 10, the main building blocks are

- Input processing,
- Interleaving and modulation,
- Frame building,
- OFDM generation.

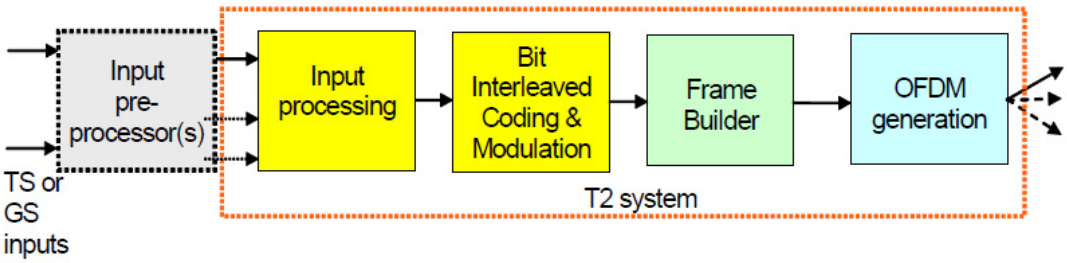

Figure 10 - High-level view of the DVB-T2 system

The incoming Transport Stream packets (generic streams are not considered in ROMEO) are packetized into Baseband frames as illustrated in Figure 11.

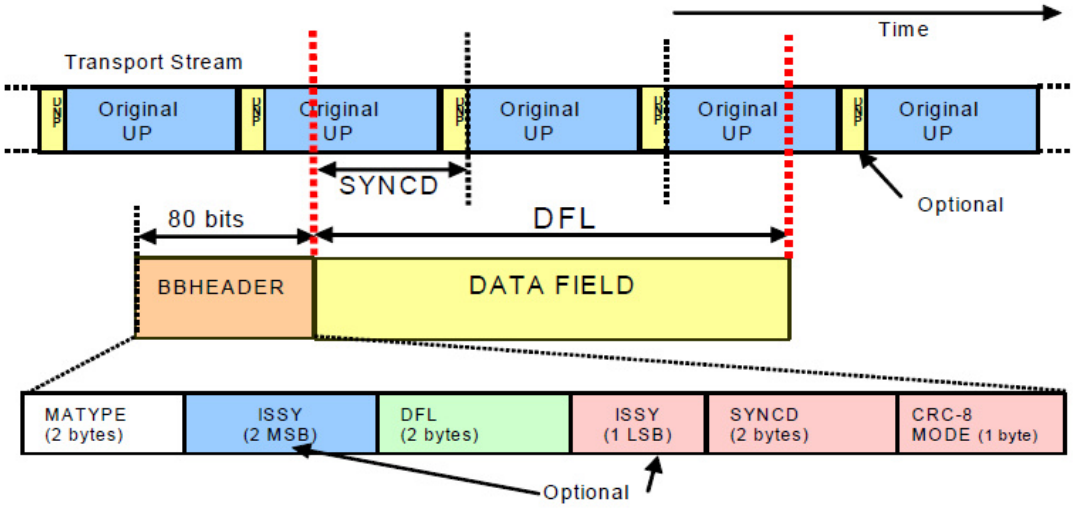

Figure 11 - Principle of Baseband Frame building in DVB-T2

The Forward Error Correction for the Baseband frames uses LDPC and BCH codes and the redundancy is added to the Baseband frame as illustrated in Figure 12 (for more details see [8]).

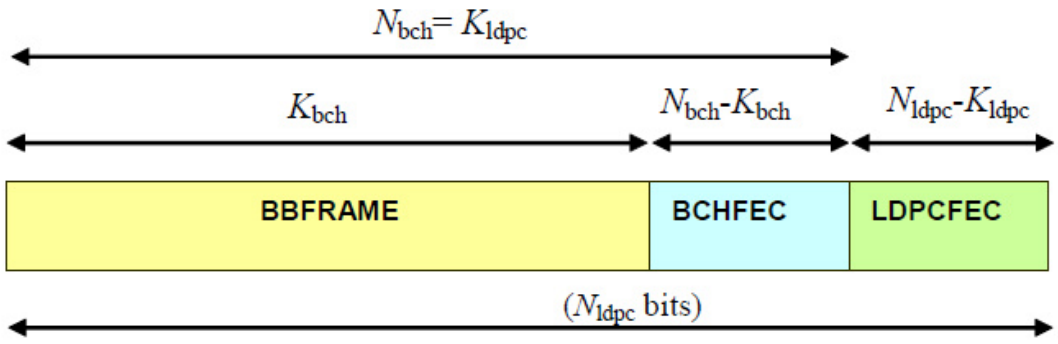

Figure 12 - FEC for Baseband frames in DVB-T2

After the FEC encoding, the frames are mapped to the different constellations selected for the DVB-T2 transmission system. To obtain an optimal power distribution over all OFDM carriers, different interleaving algorithms are implemented.

The Layer 1 (L1) signalling provides the data to the receiver that is necessary for a rapid synchronisation of the DVB-T2 demodulator and the identification of the Physical Layer Pipe (PLP) that should be demodulated and decoded.

In ROMEO, the 3D DVB signal is transported in a single PLP.

The availability of additional views/layers through IP networks needs to be signalled in the PSI/SI tables of the DVB signal. For this purpose, a new descriptor that points to a URL were the terminal could find the information on these views/ layers is proposed, e.g. in an XML file.

This descriptor is named 'Additional Content URL Descriptor' and is built after the similar descriptors for other additional DVB services, e.g. HbbTV.

The descriptor informs to the terminal that additional views are available over the IP network (P2P) and further information can be retrieved from the Internet. The identification of the additional views can make use of the following information in the DVB PSI/SI tables:

- Original Network ID
- Transport Stream ID
- Service ID (also possible: Event ID)

In the case that the URL to which the descriptor points is not changed, the descriptor can be located permanently in the SDT (Service Description Table). For a temporary association of a URL with a certain program, it can be inserted into the EIT (Event Information Table).

Descriptor Syntax

The following table defines the syntax of the proposed descriptor.

Syntax	Bits	Identifier
additional_content_URL_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
descriptor_tag_extension	8	uimsbf
reserved_future_use	8	uimsbf
for (i=0;i<N;i++){		
additional_content_id	16	uimsbf
URL_base_length	8	uimsbf
for (j=0;j<URL_base_length;j++){		
URL_base_byte	8	uimsbf
}		
URL_extension_count	8	uimsbf
for (k=0;k< URL_extension_count;k++){		
extension_id	8	uimsbf
reserved_future_use	8	uimsbf
URL_extension_length	8	uimsbf
for (l=0;l< URL_extension_length;l++){		
URL_extension_byte	8	uimsbf
}		
}		
}		

Table 4 - Syntax of proposed descriptor

descriptor_tag: The descriptor tag is an 8-bit field which identifies each descriptor. Those values with MPEG-2 normative meaning are described in ISO/IEC 13818-1 [18]. The value of the descriptor_tag is **0x7F** for the **extension_descriptor()**.

descriptor_length: The descriptor length is an 8-bit field specifying the total number of bytes of the data portion of the descriptor.

descriptor_tag_extension: The descriptor tag extension is an 8-bit field which identifies each extended descriptor. The value of the descriptor_tag_extension **has to be defined by tm-gbs**. (Suggestion: 0x12).

reserved_future_use: when used in the clause defining the coded bit stream, it indicates that the value may be used in the future for ETSI defined extensions

NOTE: Unless otherwise is specified within the present document, all "reserved_future_use" bits are set to "1".

additional_content_id: This 16-bit field uniquely identifies the additional available content to the service or event.

additional_content_id	Description
0x0000	Reserved
0x0001	ROMEO content
0x0002 to 0x0FFF	Reserved for future use
0x1000 to 0xFFFF	User defined

URL_base_length: This 8-bit field specifies the length in bytes of the following base part of the URL.

URL_base_byte: This is an 8-bit field. These bytes form the first part of a HTTP URL conforming to HTTP 1.0 (see RFC 1945 [18]), or the first part of an HTTPS URL conforming to RFC 2818 [19] or the first part of another URL conforming to RFC 3986 [14].

All bytes interpreted as text shall be encoded as UTF8, but shall not include the null character.

e.g. **„www.ROMEO.Service-Guide.tv“**

URL_extension_count: This 8-bit field indicates the number of URL extensions conveyed by this descriptor.

URL_extension_length: This 8-bit field indicates the number of bytes in the extension part of the URL.

URL_extension_byte: These bytes form the later part of an HTTP URL conforming to HTTP 1.0 (see RFC 1945 [18]), or the later part of an HTTPS URL conforming to RFC 2818 [19] or else a URL whose scheme is supported by a registered interaction channel transport service provider implementation.

All bytes interpreted as text shall be encoded as UTF8, but shall not include the null character.

URLs are formed by concatenating the URL extension with the preceding URL base. The URL so formed either identifies a file system directory or a specific XML file.

e.g. **„/romeo_esg.xml“**

Other possible additional content URL's:

http://www.youtube.com/results?search_query=obama&oq=obama&gs_l=youtube-reduced.3..0l4.14784.16444.0.21746.5.3.0.2.2.0.395.395.3-1.1.0...0.0...1ac.1.5LFwyloagHE

or

http://www.zdf.de/ZDFmediathek/hauptnavigation/startseite#/beitrag/video/257404/heute-100SEC

3 Proposed packetization schemes

3.1 Protocol stack

3.1.1 State of the art

RTP (real-time transport protocol) provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services. Those services include payload type identification, sequence numbering, time stamping and delivery monitoring.

Applications typically run RTP on top of UDP to make use of its multiplexing and checksum services. The sequence numbers included in RTP allow the receiver to reconstruct the sender's packet sequence, but sequence numbers might also be used to determine the proper location of a packet, for example in video decoding, without necessarily decoding packets in sequence.

RTP does not address resource reservation and does not guarantee quality-of-service for real-time services, but relies on lower-layer services to do so. The data transport is augmented by a control protocol (RTCP) to allow monitoring of the data delivery in a scalable manner to large multicast networks, and to provide minimal control and identification functionality. RTP and RTCP are designed to be independent of the underlying transport and network layers. Both are specified in the IETF RFC 3550[1].

There are several recommendations, as RFC 2250 and RFC 6184 [4] that specify how to encapsulate multimedia data using Real Time Protocol according to the utilized coding standard.

RFC 2250[2] describes a packetization scheme for MPEG video (and audio) stream by means of two approaches: the first one is designed to support maximum interoperability between different MPEG scenarios; the second one is aimed to guarantee maximum compatibility with other RTP media architectures. More precisely, on one hand, it proposes encapsulation for MPEG-1 system streams and MPEG2 transport and program streams with RTP; and on the other hand, it proposes an encapsulation of video and audio data (Elementary Streams) directly with RTP.

RFC 6184 [5] specifies a RTP payload format for ITU-T Recommendation H.264 video codec. It allows encapsulating one or more Network Abstraction Layer Units, generated by an H.264 codec, in a RTP packet.

Moreover, IETF proposed an RTP payload format for the Multiview Video Coding (an amendment of H.264 video codec) that deserves an in-depth analysis, since it has wide applicability, such as 3D video streaming, free-viewpoint video, and 3DTV.

3.1.2 Proposed protocol stack for ROMEO

Within the ROMEO IP delivery mechanism, TS/UDP/IP protocol stack will be utilized for the real-time media streaming. Additional capabilities for streaming are provided by P2P packetization scheme and intelligent chunk selection algorithm that enable content aware delivery over the IP network.

The internet protocol (IP) is a packet-based-network transport with a header containing addressing and control information for packet routing through packet-switching networks. In the case of streaming over IP networks, multiple protocols, such as UDP (described below),

may be carried in the IP payload, each with its own header and payload that recursively carries another protocol packet [9].

The service provided by UDP is an unreliable service that provides no guarantees for delivery and no protection from duplication. It provides a minimal, unreliable, best-effort, message-passing transport to applications and upper-layer protocols. Compared to other transport protocols, UDP does not establish end-to-end connections between communicating end systems. It is a stateless protocol, which makes it suitable for very large numbers of clients. Because of these characteristics, UDP can offer a very efficient communication transport to some applications, such as in streaming media applications like IPTV. Another characteristic of UDP is that it provides no inherent congestion control mechanism or reliability. So, applications requiring reliable delivery need to be designed responsibly. With all of these characteristics, the simplicity of UDP reduces the overhead from using the protocol and the services may be adequate in many cases, especially the ones requiring streaming.

The protocol stack explained above is preferred to the common TS/RTP/UDP/IP scheme since the proposed packetization and chunk selection mechanisms also provides the timing information, payload identification and sequence numbering capabilities which are the main advantages of the alternative approach without 12-bytes extra overhead caused by RTP header.

3.2 Packetization schemes

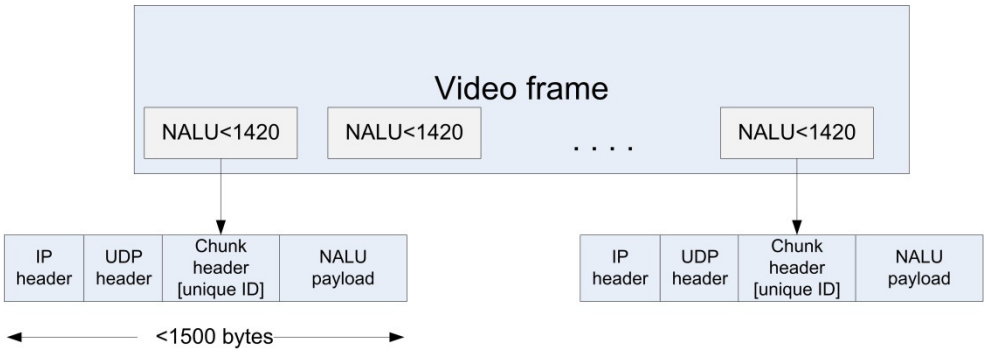

Figure 13 - Packet structure for transmission over IP

Since ROMEO employs a hybrid delivery system which utilises both DVB and IP for 3D content delivery, DVB-IP media synchronization is a key point for the quality user experience. Since DVB-T2 standard will be used for DVB broadcasting and it is using MPEG2-TS packets, using the same encapsulation for P2P has an advantage of having a common standard that can be handled by the same tools at the receiving side. In addition, instead of carrying the TS packets in the payload without any specific knowledge about the content of the packets, ROMEO P2P packets are formed by encapsulating the TS packets with all of the necessary information about the content it has within the chunk header as can be seen in Table 3. With the employed packetization scheme, each chunk will carry all the information needed for content aware P2P delivery, chunk selection and retransmission mechanisms. In this way, the data delivery performance is improved since components do not need to extract any specific information within the chunk throughout the streaming over P2P networks. Moreover, since chunk payloads are formed to include MPEG2-TS packets that correspond to a single NAL unit (which are self-contained and decodable units) as can be seen in Figure 13 and explained in section 3.3.1, one lost packet will typically result in only one lost NAL unit, not more. In

addition, with the help of the unique chunk identifiers, it is possible to detect this missed chunk and request it from the sender with the intelligent chunk selection mechanism. Furthermore, formation of the small-sized (<MTU size) chunks also improves the performance of the chunk recovery process, since with the employed scheme, it is possible to request a small sized single NALU with its unique chunk id.

3.3 Media partitioning and muxing

3.3.1 Multiview video streams

The structure of the formed SVC elementary stream used in ROMEO is shown in Figure 14. While creating the MPEG-2 TS multiplex, the parameter set NALUs (as well as other associated SVC extensions of parameter sets and other SEI NALUs) should be repeated in front of each Group of Pictures (GOP), where each GOP starts with I-frame.

In Figure 14, all SVC related NALUs are tagged accordingly. Note that each 'base layer VCL NALU' is preceded by a special Prefix NALU that is an SVC related packet. On the other hand, 'enhancement layer VCL NALU' is not preceded by such special packets. Also note that some frames can consist of multiple base layer and enhancement layer NALUs thanks to the multi-slice encoding. Since in ROMEO it has been decided to insert a single NALU in each chunk and that each chunk's size should not exceed the MTU (i.e., 1500 bytes), it is a must to create multi-slice frames (see Figure 14). Therefore, especially I-frames will consist of many such base and enhancement layer NALUs, each to be introduced into a separate packetized elementary stream (PES) packet. However, some frames, especially P-frames can consist of a single base layer (preceded by Prefix NALU) and a single enhancement layer NALU packet.

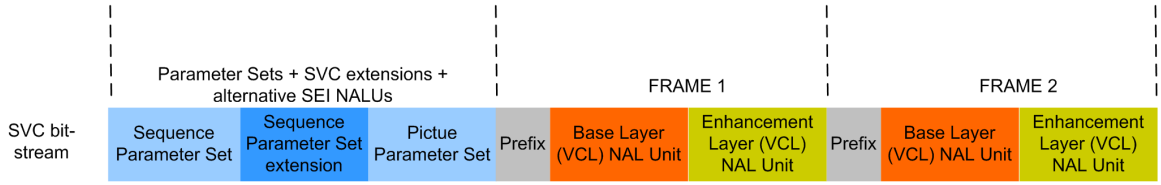

Figure 14 - SVC bit-stream structure (single slice)

Unlike AVC, the SVC bitstream will be packetized in several PES, targeting several PIDs in the MPEG-2 TS, with one PID per layer as depicted in Figure 16. This allows first backward compatibility for the base layer with MPEG-2 TS packetization of AVC, and allows also selecting the number of enhancement layer to be decoded. At the output of the demultiplexer, the different NAL units have to be re-assembled in a single bitstream, to be sent to the SVC decoder.

For AVC video streams and base layer of SVC, the first PES_packet_data_byte following the PES header shall be the first byte of an AVC access unit or the first byte of an AVC slice. For SVC enhancement layer video streams the first PES_packet_data_byte following the PES header shall be the first byte of an SVC dependency representation or the first byte of an SVC slice.

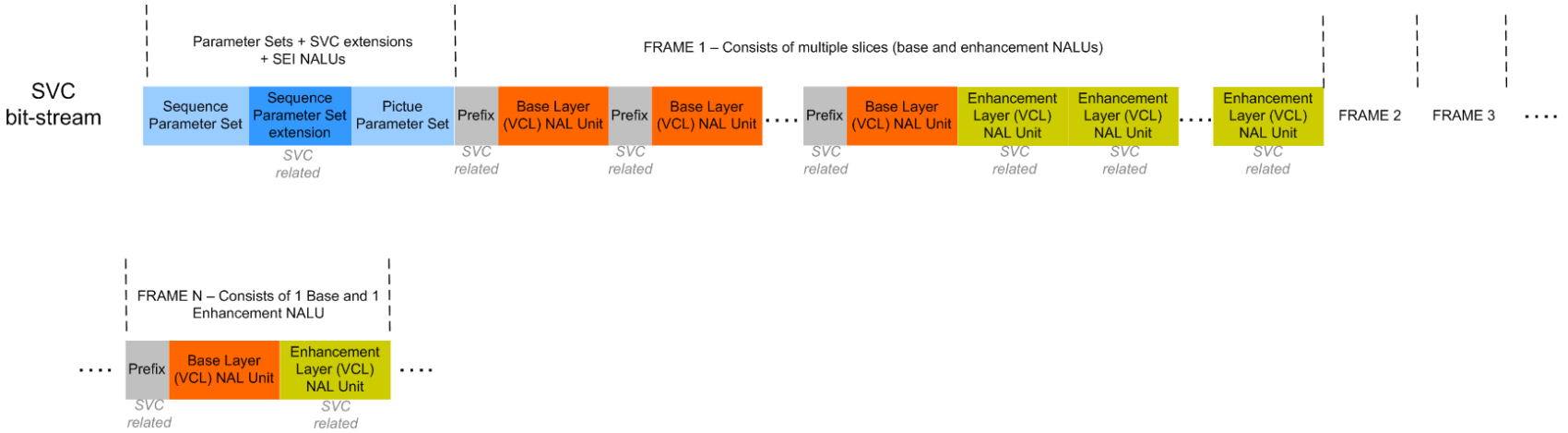

Figure 15 - SVC bit-stream structure (multiple slices)

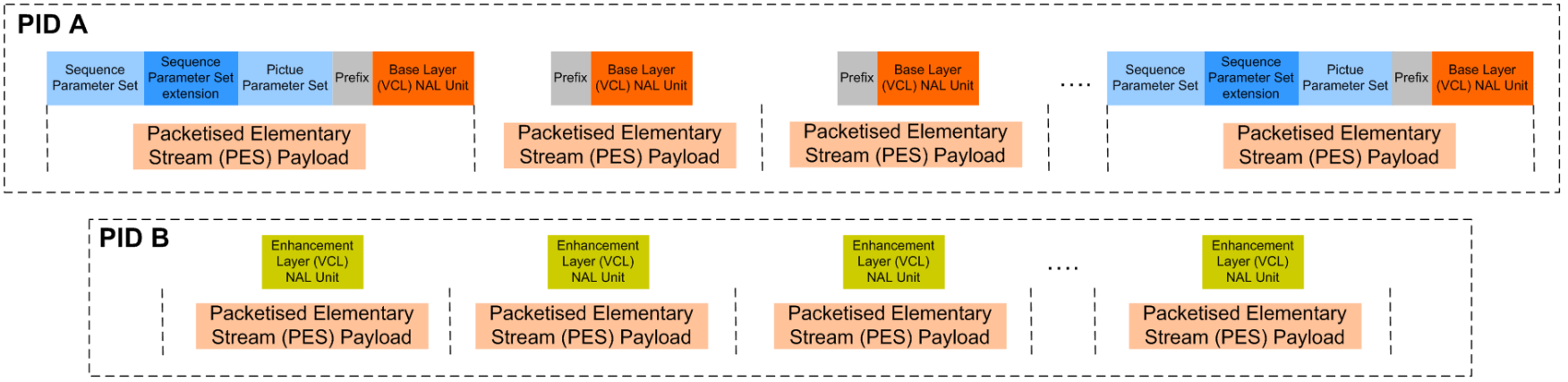

Figure 16 - Packetized Elementary Stream (PES) formation

As the different layers of SVC will be separated into MPEG-2 TS packets of different PIDs, at the demultiplexer stage, a single bitstream has to be re-created by the concatenation of the several layers (access units reassembling),before being sent to the SVC decoder.

Figure 17 shows how the PES packets are inserted into the TS packets (each fixed length – 188 bytes) and how they form the chunk payload.

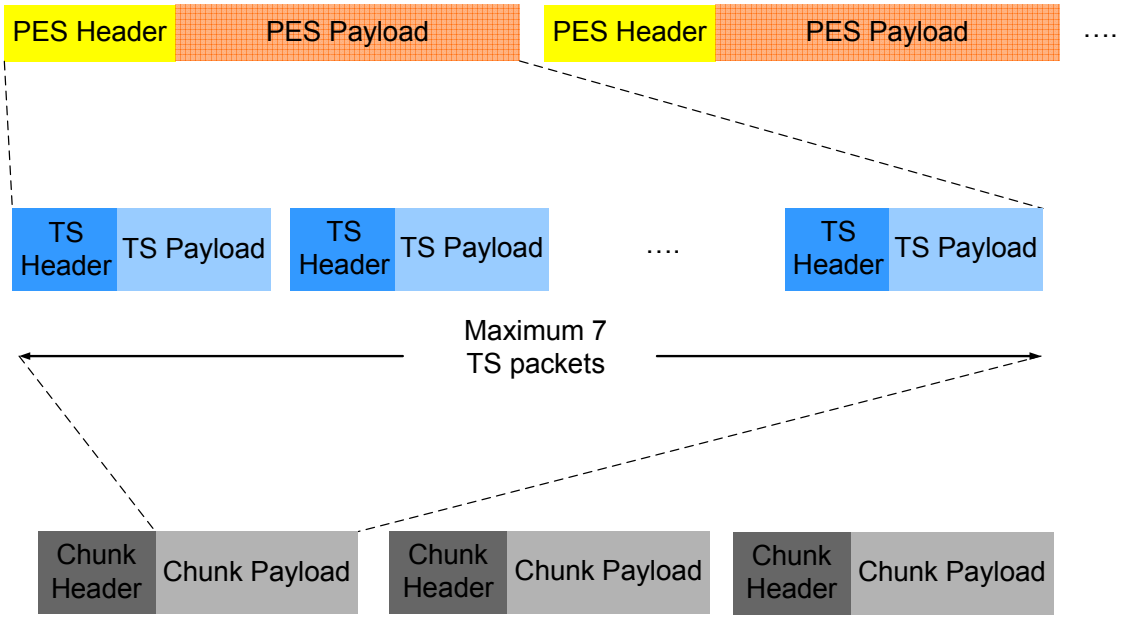

Figure 17 - PES to TS to chunk

Since the IP transmission format follows a structure like 'chunk → UDP → IP' and the max IP packet size should not exceed 1500 bytes to avoid fragmentation in the network layer for the sake of decreased delay, the maximum NALU packet size is set around 1200 bytes in the encoder. This takes into consideration that a maximum of seven TS packets can enter a single chunk (7*188 = 1316 bytes) and the TS and PES packet headers as well as other chunk and UDP headers. This is still open to further adjustments, if the final packet size under certain circumstances is likely to exceed the MTU size.

3.3.2 Metadata streams

Metadata content and format, to be used in ROMEO (camera information and calibration) are described in Deliverable D3.1 (section 2.3.3 and Appendix B). These metadata use SMPTE336M / SMPTE RP210 format and dictionary. These metadata are transported over MPEG-2 TS though PES, accordingly to section 2.12.4 of ISO/IEC 13818-1:2007, "Use of PES packets to transport metadata". This multiplexing method allows transporting metadata synchronized with the elementary stream.

The fine synchronization between metadata and elementary stream(s) is done through PTS information. When metadata is sampled at the same time as a frame the metadata and the frame will have the same PTS. If the metadata is not sampled at the same time as the frame, it will have a different PTS, but will use the same timeline as the frame.

The payload of such PES packets will consist in a collection of Metadata Access Units as illustrated in Figure 18. A metadata Access Unit Cell contains a 5-byte header.

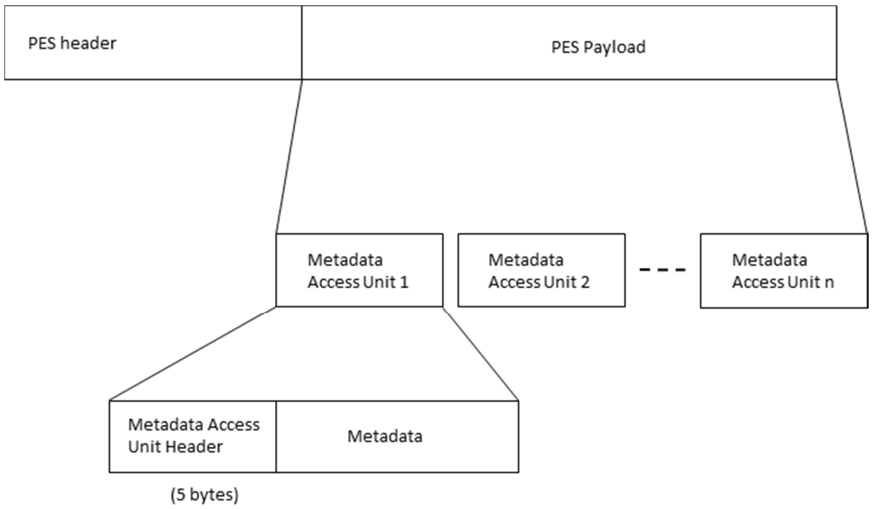

Figure 18 - Synchronous Metadata PES Stream

Requirements for synchronous SMPTE metadata multiplexing are:

- The stream_id value in the PES header shall be 0xfc, indicating “metadata stream”.
- Each PES packet shall have a PTS to be used to synchronize the metadata with the video elementary stream. The PTS in the PES header applies to each Access Unit contained in the PES packet. No DTS shall be coded
- In each PES packet that carries metadata, the first PES packet data byte shall be the first byte of a Metadata Access Unit Cell.
- When inserting synchronous metadata into a transport stream which already carries synchronous metadata new metadata shall be added to the existing synchronous metadata stream.

In Program Map Table (PMT), The Metadata Stream shall be defined as a separate stream within the same Program as the video and audio elementary stream, with a stream_type of 0x15, indicating “Metadata carried in PES packets”.

3.3.3 Audio streams

3.3.3.1 Spatial audio stream

Similar to the video streams, the audio Elementary Streams (in MPEG-4 ADTS format) described in section 2.1.3 are further packetized into PES streams. The structure of an audio PES packet is shown in Figure 19.

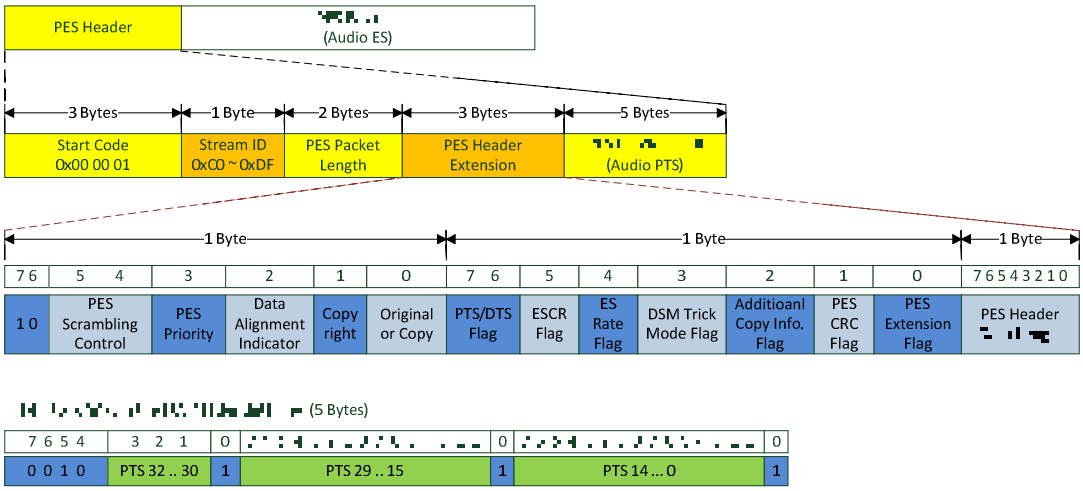

Figure 19 - PES packet format for audio encapsulation in Romeo

For audio PES packet, the Stream ID can be chosen from 0xC0 to 0xDF and the extension field is always presented immediately after the PES header length field.

The PES header extension includes various fields to indicate extra information or feature presented for the packet. Among them, the audio decoder and renderer are particularly interested in the PTS information which is used to synchronise the audio with video. The PTS/DTS flag in the header extension field can be set as following:

- 00: no PTS or DTS data is presented
- 01: not allowed
- 10: only PTS is appended to the header data field
- 11: both PTS and DTS are appended to the header data field.

For the sake of simplifying client audio decoding process in Romeo, only PTS extension is used in the audio stream PES packet and each PES packet is required to include the PTS extension field. No other extensions were added in the PES audio stream. Therefore the PES/DTS flag in the header extension is set to “10” and the PES Header Data Length filed is set to 5, which is the length of PTS data in byte.

Other flag fields (except the fields with extension data) can be set as required.

Although the maximal size of a PES packet is 64K bytes, the actual number of audio ADTS frames in one PES packet should not be too large. Given that each ADTS frame in the audio Elementary Stream is the compression of 1024 audio samples and the sampling rate is 48000Hz, the ADTS frame time is about 21.3333ms. Assuming the audio decoder can buffer about 100ms audio data, the number of ADTS frames in one audio PES packet should not be more than 5.

Taking into account the aforementioned considerations, the constraints of the ROMEO audio stream packetization are listed as following:

- Audio Elementary Stream: MPEG-4 ADTS audio frames of multi-channel audio.
- All audio PES packets should include the PTS extension with PTS/DTS flag being set to “10” in the header.
- No other PES header extensions are included. Other flags in the PES header should be reset to zero.

- No more than 5 audio ADTS frames are packaged in one PES packet.
- The audio PES will occupy one PID in the multiplexed transport stream.

The audio PES stream is further multiplexed with other PESs (e.g. video PES) into a transport stream for DVB-T and IP delivery.

3.3.3.2 Spatial audio scene streaming

The spatial audio description is a set of Boolean, integer, String, float and matrix attributes that may change over time, either abruptly or animated. This is also true for animated Vector Graphics that have been standardised in MPEG as MPEG-4 Part 20 Lightweight Application Scene Representation (LASeR), therefore this mapping is heavily inspired by the mechanisms used in MPEG LASeR.

The 8-byte header of the MPEG-PES Mapping for spatial Audio Description is defined as following:

```
|0 1 2 3 4 5 6 7|0 1 2 3 4 5 6 7|0 1 2 3 4 5 6 7| bytes 3 – 7 |

 R* Sequence No *    FLAGS       * Reserved  *Timestamp (42bit)*
```

R – Refresh bit. If set indicates a tune-in point to the scene – no changes are made to the scene. A receiver should ignore these *refresh scenes* if in a valid state. Otherwise it is an *update scene* containing new information.

Sequence No – increments with each non-refresh packet. The receiver can detect packet losses and set the state to invalid, to resync at the next refresh scene

FLAGS – 0: Delete all, 1: BinaryFormat (n/a) 2: gzip 3-7: future extensions

Reserved – for future extensions

Timestamp – 33+9bit timestamp compatible to DVB (27MHz) to create a scene timeline

Packet length is given in PES header (the 16bit limitation should not be an issue). Timing information is given in TS Adaptation Field to align audio scene timeline with programme PCR.

The mapping to PES is mostly agnostic to the actual format of the audio description language but will be SpatDIF/OSC with an extension to simply modify single rows.

A Refresh Scene is a regular OSC file while an Update Scene pre-pads each line with either ‘I’, ‘D’, or ‘U’ for insert, delete and update respectively. It is thus mandatory that each line is unique, but this is the case for SpatDIF. All timing will be based on the spatial audio scene timeline which is defined through the 27MHz timestamp header and can be synchronized to the main programme's PCR. Links to external audio resources should be referenced as URI, either dvb-uri for audio streams embedded in DVB or regular HTTP or ROMEO P2P URIs.

Implementation:

Server:

The Server implementation (TS Generator) takes either a complete offline scene as input or as attached to a live spatial audio scene composer.

Configurable parameters are the start of the timeline, binary and gzip flag, as well as the refresh-scene interval in ms, prefetch time. All scene elements are then sorted by their change time (0 if constant). Now a first scene is generated for time 0 where all later elements are removed. Then for each step it is checked if there is any update prior to the next *refresh scene*. If there is none, an identical scene is copied into the next *refresh scene* packet, just with an updated timestamp. If there is an update, a new *update scene* is generated and the modified *refresh scene* is prepared. For updates introducing new sources it is possible to configure a prefetch time. That way, these elements will be added to the scene in advance to allow for prefetching from the network.

For the live version, also a constant bitrate option is implemented with configurable bitrate. This should be planned with enough overhead. For the offline version constant bitrate can be better generated afterwards.

Client:

The Receiver implements a state machine with 2 states:

- *Invalid* (Starting state): Wait for and then process *Refresh Scene* – set state to *Valid*.
- *Valid* (Working State): Ignore *Refresh Scenes*, Process *Update Scenes*, when one or more Update scene is lost set state to *Invalid* (detect by SequenceNo).

Processing the scene means gunzip (if flag is active) or binary decompress (not implemented) and then sending the scene with the respective API call to the spatial audio module.

3.3.4 Synchronization and buffer management

Before entering the audio and video decoding modules, the different streams will be processed by an “AV Sync & Demux & Buffer” module. Basically, this module is responsible for:

- Buffering audio streams and video-view streams. The buffer will hold several seconds of data (configurable and adaptable) before streaming it to the clusters. This is done to prevent buffer underrun and to compensate different arrival times from different peers.
- Synchronising the audio and the video; the streams of the video views; DVB-T2 and P2P streams.
- Sending the synchronised streams to the video and audio decoders. The streams should be sent several frames before the indicated presentation time, in order to allow the decoders enough time to decode before sending the frame for rendering.
- Combining the base-layer and the enhancement-layer of video views.

In ROMEO, the PCR information is not transmitted through TS streams for IP transmissions.

Under certain circumstances, a ROMEO receiver may use the DVB-PCR for clock frequency generation and do the synchronization (buffer management) between DVB and IP with PTS/DTS only.

For instance, considering the video transmission over IP, PCR might be not necessary if the source is prepared with following rules:

- All videos have identical GOP structures,Section 4 is a focus on the reliability and error resilience aspects
- All slices belonging to the same picture have identical PTS/DTS values.

In any case, for synchronization purposes at client side, a PCR field is defined in the IP chunk header (see Figure 13). The P2P contains the DTS and PTS for the additional views (or audio channels) and layers in the PES packets. Hence, for each video frame, they are identical in the DVB signal and the P2P signal. This will enable the terminal to synchronise the time stamps of the views/ layers received over the P2P network with the DVB signal. The buffer size at client side needs to be large enough to buffer out the latency between networks plus jitter as illustrated in Figure 20.

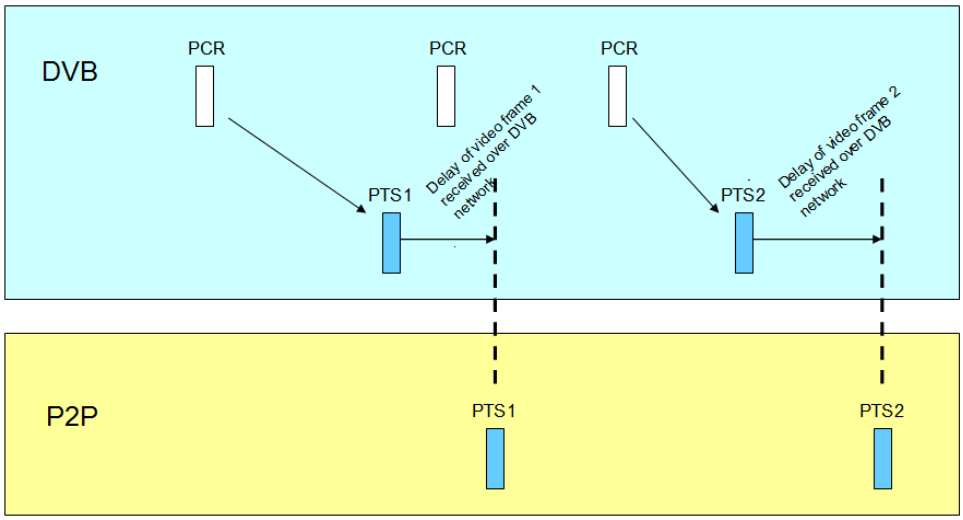

Figure 20 - Timing between DVB and P2P if the delay is longer in the P2P network

4 Adaptation and error resilience

4.1 Media aware mechanisms

The Media Aware Proxy (MAP) is one of the three sub-modules of the ROMEO mobility component. MAP is a transparent user-space module used for low delay filtering of streams available to the mobile users, as shown in Figure 21. The need for such a middle box derives from the fact that each mobile user will be able to handover from one access network to another without losing the QoE due to different network capabilities.Therefore in case of a handover to a new network, MAP will either drop or forward packets that carry specific views to the receiving user. Its functionality extends from the Network Layer to the Application Layer providing video stream adaptation by taking into account the clients' network conditions. Each received packet is forwarded to the routing module ("*ip_queue*") and its header is parsed in order to identify the information without changing the header's fields. It is important to underline the fact that the whole process is running in kernel level, hence it is very robust and contributes a negligible amount of time to the overall end-to-end delay.

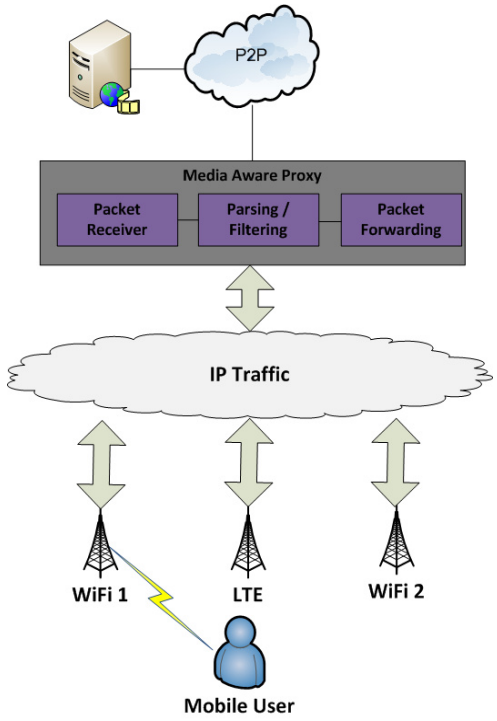

Figure 21 - Traffic through Media Aware Proxy

The main processes of MAP are described in Figure 22. The flow chart illustrates three processes that handle the flow of packets and the bit rate adaptation. The "*map_thread"* is the main map process which overrides the router's normal routing procedures. To do so, the "*map_thread*" binds to *Netlink* socket and receives the IP packets containing the P2P Chunks, exploiting the libraries of *"ip_queue*" module (*libipq.h*) and the *Netfilter* mechanism (*netfilter.h*). It identifies the video information by parsing the chunk headers and based on the "*adte_check_packet()*" method, which is explained in more detail in Section 4.1.1.2, it forwards or drops the packets. Moreover, it updates the views/layers structure in order to indicate the number of the received views/layers.

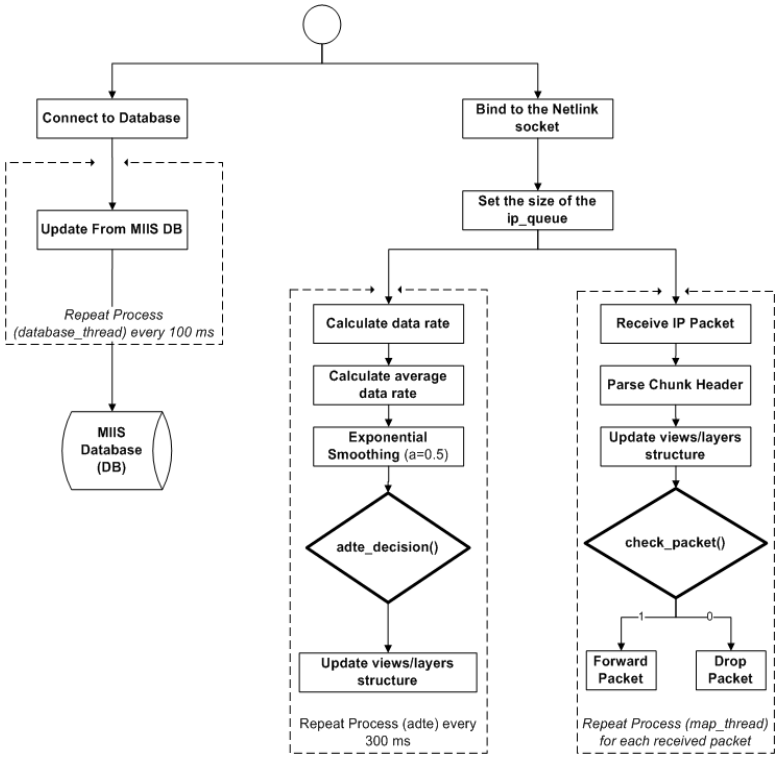

Figure 22 - Abstract MAP Flowchart

4.1.1 Packet discarding mechanisms for mobility component

The packet discarding mechanism for mobility component is called Adaptation Decision Taking Engine (ADTE) and it is a fundamental part of MAP. ADTE comprises two processes, called *adte_decision()* and *adte_check_packet()*. The first periodically decides which layers will be dropped based on the current available bandwidth and the second checks if an incoming packet should be dropped or forwarded.Below, both processes are further described.

4.1.1.1 *adte_decision()*

The *adte_decision* utilizes two network metrics, the client's throughput, which is estimated from the network based on passive throughput estimation schemes and the predicted data rate, which is calculated using the Simple Exponential Smoothing Model [10]. There are two *adte_decision* schemes developed for the MAP module differing in the decision of which layer should be first dropped.

- The first scheme ensures higher priority to the first stereo view by dropping the higher enhancement layers of the second stereo view.
- The second scheme provides higher priority to the lower layers of both stereo views, by selecting to drop the higher layers of both views.

A comparison of the two implemented schemes indicates that the first ensures higher quality for the first stereo view, while the latter provides the same level of quality for both stereo views.

In Figure 23 a more detailed flowchart of the first *adte_decision()* method is shown. In the case where the predicted transmission data rate is higher than the available bandwidth, *adte_decision()* searches for the highest available enhancement layer of the second stereo

view that can be dropped due to no decoding dependencies to other layers. It is shown that the enhancement layer of the first stereo view will only be dropped after all the enhancement layers of the second view have been already dropped.

On the other hand, if the sum of the predicted transmission rate and the data rate of the dropped layer is lower than the available bandwidth, it means that the dropped layer can be added to the flow and transmitted to the client. The basic difference of this scheme compared to the next decision mechanism is that all the enhancement layers of the first stereo view must be forwarded in order to re-forward an enhancement layer of the second stereo view.

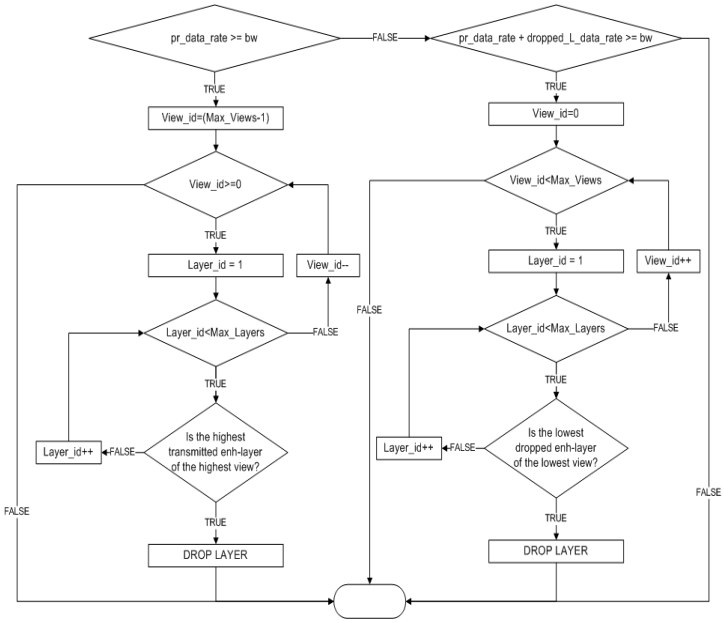

Figure 23 - First adte_decision method

Figure 24 shows the flowchart diagram of the second *adte_decision()* method. In case where the predicted data rate is higher than the available bandwidth, *adte_decision()* searches for the highest available enhancement layer of the right view that can be dropped. As it is shown, in order to drop an enhancement layer of the first stereo view the corresponding layer of the other view must have been dropped first.

On the other hand, if the sum of the predicted data rate plus the data rate of the dropped layer is lower than the available bandwidth, it searches for the lowest layer of the first stereo view that can be added again in the flow. Between the same enhancement layers of both views, the enhancement layer of the first view will be added to the flow first.

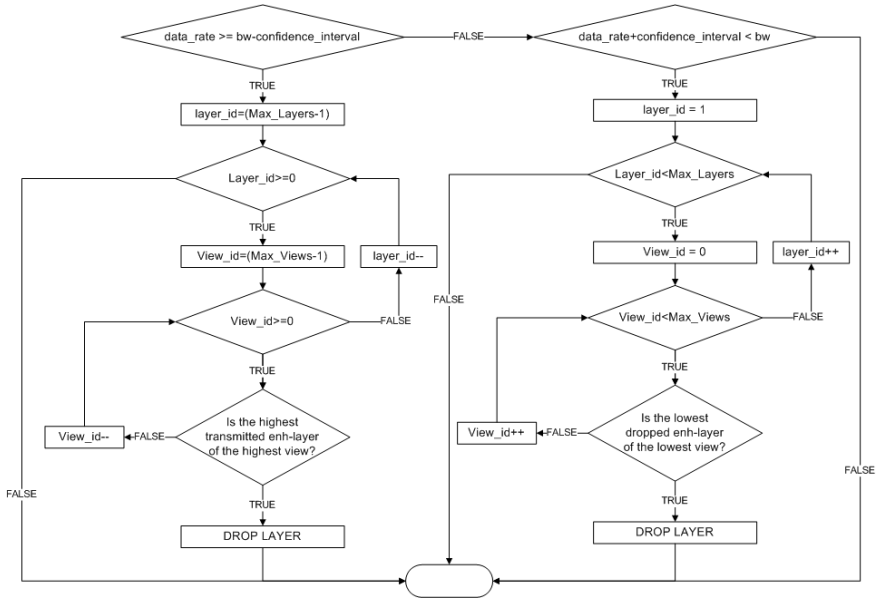

Figure 24 - Second adte_decision method

4.1.1.2 *adte_check_packet()*

Figure 25 presents a flowchart diagram of the *adte_check_packet()* method which is responsible to make a decision on whether a chunk is going to be forwarded or dropped based on the *adte_decision()*.

A packet is going to be forwarded either if it is unknown (i.e. is not a video packet) or if the chunk is part of a video layer that must be forwarded. On the other hand, a chunk is going to be dropped if it encapsulates payload of a video layer that is to be dropped (based on *adte_decision()*).

In case where the layer has been previously dropped, while the new decision is to be added to the flow, all the chunks up to the first chunk of the next GOP (i.e. new I-frame) are also going to be dropped. Moreover, in case where the layer was previously forwarded, while the new decision is to be dropped the chunks up to the next frame of the layer are also going to be forwarded.

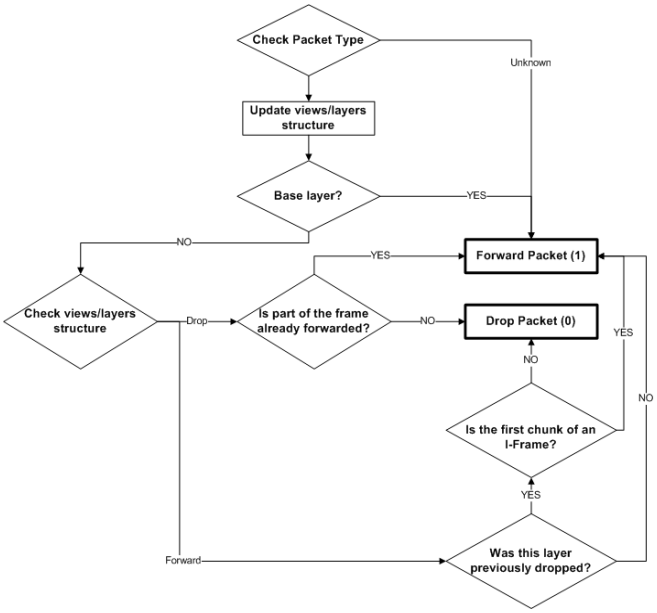

Figure 25 - adte_check_packet method

4.1.2 Experimental Setup

In order to evaluate the performance of the aforementioned adaptation mechanisms, a testbed is developed to enable real-time transmission of video streaming data. The main building blocks as well as, the network topology of the test-bed are illustrated in Figure 26. The MAP module has already been described, while the rest of the tools and implemented modules are summarized below.

- Chunk Streamer - The Multi-view H.264/SVC Streamer has been implemented in order to handle the packetization and smooth transmission of the multiple layered encoded views. Multiple threads are executed in order to handle the packetization and transmission procedure of the different views by creating concurrent UDP/IP connection to the client. Each thread is responsible for handling the chunk packetization procedure and the time efficient transmission of each packet in order to fulfil the frames per second rate.
- Chunk Client - The Multi-view H.264/SVC Client is able to receive multiple layered encoded views from the streamer. Each executed thread is responsible for the reception, de-jittering/buffering and depacketization of each stream. Finally, the last thread handles the synchronization of the received packets of the different views.
- Dummynet - Dummynet [11] is an open source network emulator, which emulates queue and bandwidth limitations of real networks as well as, predefined delays and packet losses that trigger the transmission bit rate adaptation mechanisms.

The experimental setup is illustrated in Figure 26.

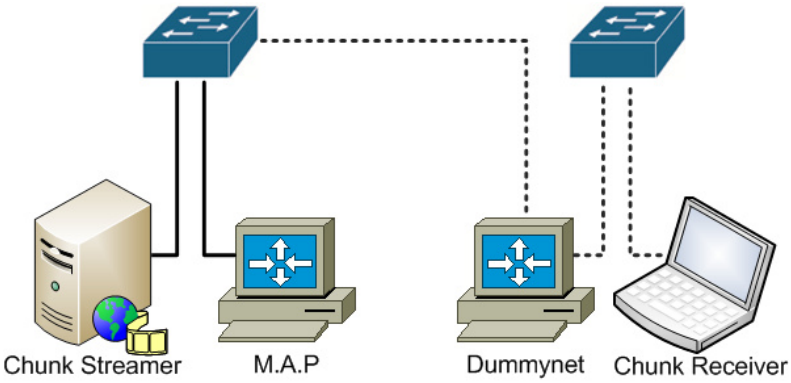

Figure 26 - Testbed Components and Topology

4.1.3 Experimental Results

This section includes a description of the experimental results obtained during streaming SVC encoded side by side video over MPEG2-TS/P2P Chunk/UDP network. The testing sequence is *"Martial Arts"*, which is a high motion sequence with texture variation and standing camera, at a resolution of 1920x1080 pixels and 25 frames per second. Medium Grain Scale scalability is applied to produce a base and one enhancement layer. The testing video sequence is transmitted over an IP network by the P2P chunk streamer that is designed to encapsulate Chunks created based on the format described in Section 3, into IP datagrams with a variable size to fit within MTU of size 1500 bytes. In order to emulate different channel bandwidths ranging from 6Mbps to 7 Mbps and to stress the performance of the system, Dummynet is set up between the Chunk streamer and the client.

The experimental scenarios include two modes of operation. The first one assumes that there is no kind of transmission rate adaptation and as a result losses occur randomly to the base and enhancement layers, when the transmission rate exceeds the available bandwidth. The second scenario demonstrates the MAP first adaptation method of Figure 23, according to which, in case of packet loss, the ADTE forces the losses to occur primarily at the enhancement layer. The following figures illustrate the obtained PSNR per frame for both experimental scenarios under different channel bandwidths (B/W).

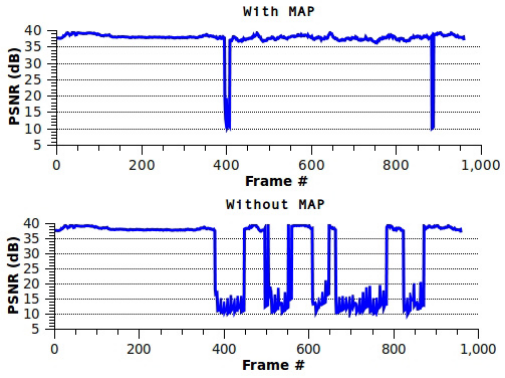

Figure 27 - Comparison of PSNR per frame of SVC side-by-side sequence obtained with and without enabling MAP (6Mbps B/W)

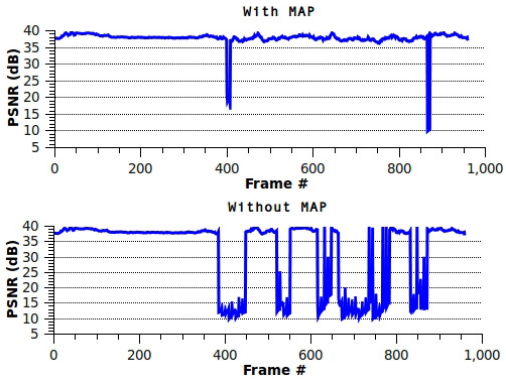

Figure 28 - Comparison of PSNR per frame of SVC side-by-side sequence obtained with and without enabling MAP (6.25Mbps B/W)

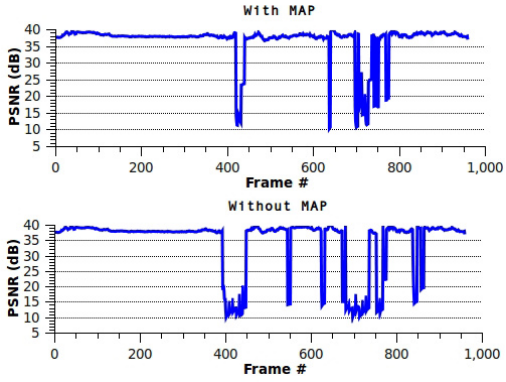

Figure 29 - Comparison of PSNR per frame of SVC side-by-side sequence obtained with and without enabling MAP (6.5Mbps B/W)

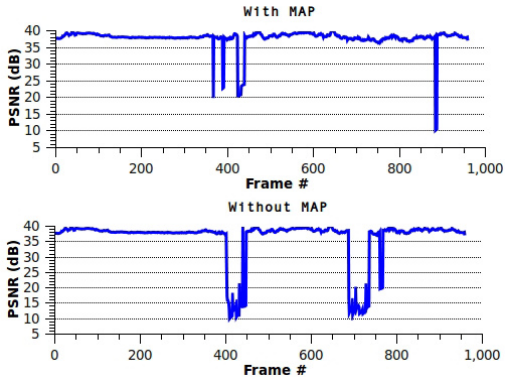

Figure 30 - Comparison of PSNR per frame of SVC side-by-side sequence obtained with and without enabling MAP (6.75Mbps B/W)

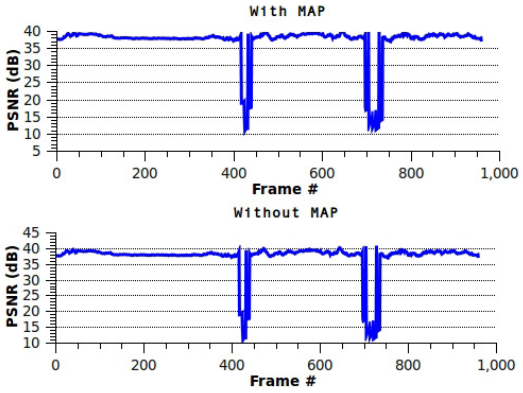

Figure 31 - Comparison of PSNR per frame of SVC side-by-side sequence obtained with and without enabling MAP (7Mbps B/W)

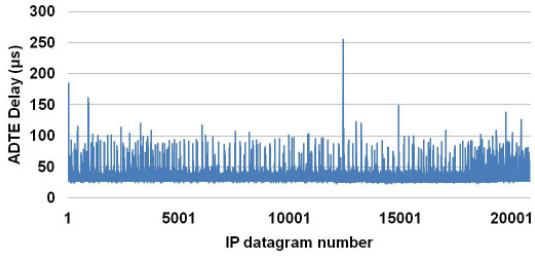

Figure 32 - Delay per packet introduced by the Adaptation Decision Engine of MAP during the simulation (6.5Mbps B/W)

From Figures 27 to 31, it is apparent that MAP succeeds to improve the PSNR of the received video significantly, almost 4dB, as shown in Table 5, compared to streaming the video without applying any kind of transmission rate adaptation. Particularly, for the scenarios of low bandwidth, the proposed adaptation scheme ensures smooth perceived quality by dropping the enhancement layer as soon as the exponential smoothing prediction algorithm indicates that the transmission rate will exceed the available channel bandwidth.

Available Bandwidth (Mbps)	With MAP	Without MAP
6	37,43 dB	29,23 dB
6.25	37,54 dB	30,95 dB
6.5	36,50 dB	33,94 dB
6.75	37,66 dB	35,74 dB
7	36,85 dB	36,84 dB

Table 5 - Average PSNR obtained from the different experimental scenarios

Moreover, Figure 32 illustrates the delay that the decision making algorithm introduces to the overall end-to end delay. Although the implementation of MAP is not yet finalized, hence there is still room for optimizing its functions and processes, it can be noticed that the added average per packet delay of 32μs is extremely small. Such a small processing delay is

explained by the fact that the ADTE process is implemented as Layer 3 hence, all the process is handled in the kernel. Therefore, it can be concluded that the development of MAP with a more intelligent and optimized rate adaptation mechanism will not impose delay related problems to the overall ROMEO end-to-end delivery system.

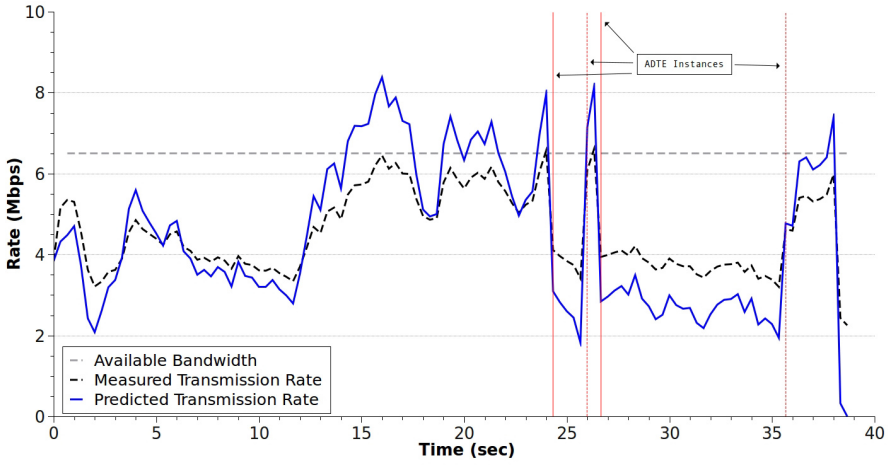

Figure 33 - Comparison of measured (actual) versus estimated transmission rate over simulation time and designation of the ADTE triggering instances (6.5Mbps B/W)

Regarding the performance of the studied ADTE adaptation process in terms of real time transmission rate estimation and adaptation, Figure 33 provides an overview of the MAP's performance in the case of 6.5Mbps B/W experimental scenario. It can be seen that the solid line, which represents the estimated rate with which the ADTE process will base its decision for adaptation is close correlated to the actual transmission rate with which the MAP module will forward the packets to the client. In particular, ADTE exploits a simple exponential smoothing mechanism for forecasting the transmission rate every 300ms and based on this forecasting it decides whether to drop the next packets of the enhancement layer or add them back to the packet flow. Forecasting the transmission rate of the MAP is a key element in the implementation and further study and optimization is undergoing. Nevertheless, initial results as in Figure 33, indicate very small prediction errors. Furthermore, this figure shows the instances when ADTE is triggered to make adaptation decisions. Specifically, the vertical solid lines indicate layer drop, while the dotted vertical lines indicate the addition of the layer to the flow.

4.2 Application based resilience mechanisms

4.2.1 Multiple description mechanisms

In ROMEO multimedia transmission over P2P-IP network, several error resiliency mechanisms exist all of them contribute to the maintenance of the overall QoE at the client side. Splitting each coded video frame into separate slices and mapping each such slice into a separate P2P chunk is a means of error resilience. This way, if a packet is hit in the network, only part of the decoded video frame will be affected. On top of this, retransmission mechanisms based on packet priority that can be triggered in the P2P transmission protocol can help reduce the visual impact of packet erasures.

In order to augment the multiple delivery paths (multiple trees), multiple descriptions are formed, each of which are individual SVC streams with multiple quality layers. Each description is sent to the client through a different multicast tree, such that when one of the

trees is congested and the packets of the corresponding description are lost, then the associated packet in the other description is used to reconstruct the video frame with a slightly downgraded quality. There are different multiple description coding and streaming techniques in the literature, all of which come at the expense of varying levels of added redundancy, computational complexity and reconstruction performance. Those techniques can be categorised roughly into three groups: methods that are based on pre-encoding (e.g., multi-phase spatial and temporal subsampling), methods integrated with the encoder (e.g., multiple description scalar and vector quantisers, correlating transforms) and methods that are based on post-encoding processing (e.g., Forward Error Correction – FEC mechanisms). Among these, methods based on pre-encoding and partly methods that are based on post-decoding processing allow us to re-use standard video codecs used in broadcasting and IP video streaming applications (e.g., backward-compatibility). FEC mechanisms comprise an efficient option for error resilient video communication in packet networks. They can perform optimally in case the channel behaviour is estimated precisely (e.g., dynamic channel erasure rate). A P2P network on the other hand exhibits a mismatched condition for FEC, where it is preferable to have prior knowledge on the network state to insert optimal redundancy. Peer instabilities and in turn dynamic multi-tree reconfigurations would hinder the performance of the anticipated FEC scheme, unless the added redundancy is controlled in the content server/ or super peers dynamically throughout the 3D video streaming. This would induce extra complexity for the server. Pre-encoding based multiple description generation on the other hand has no dependency on the network information and its design is not affected by the prior knowledge of the network state. Despite its relatively increased amount of redundancy with respect to other two categories, it facilitates the deployment of conventional SVC encoder-decoder pair with negligible computational complexity overhead.

Among the pre-encoding based methods suitable in the SVC context, several multi-phase spatial subsampling based multi-description generation schemes have been initially investigated in terms of the perceptual stereoscopic reconstruction performance under various network packet loss rates (3% and 10%). Figure 34 shows various tested multi-phase spatial subsampling based description generation techniques.

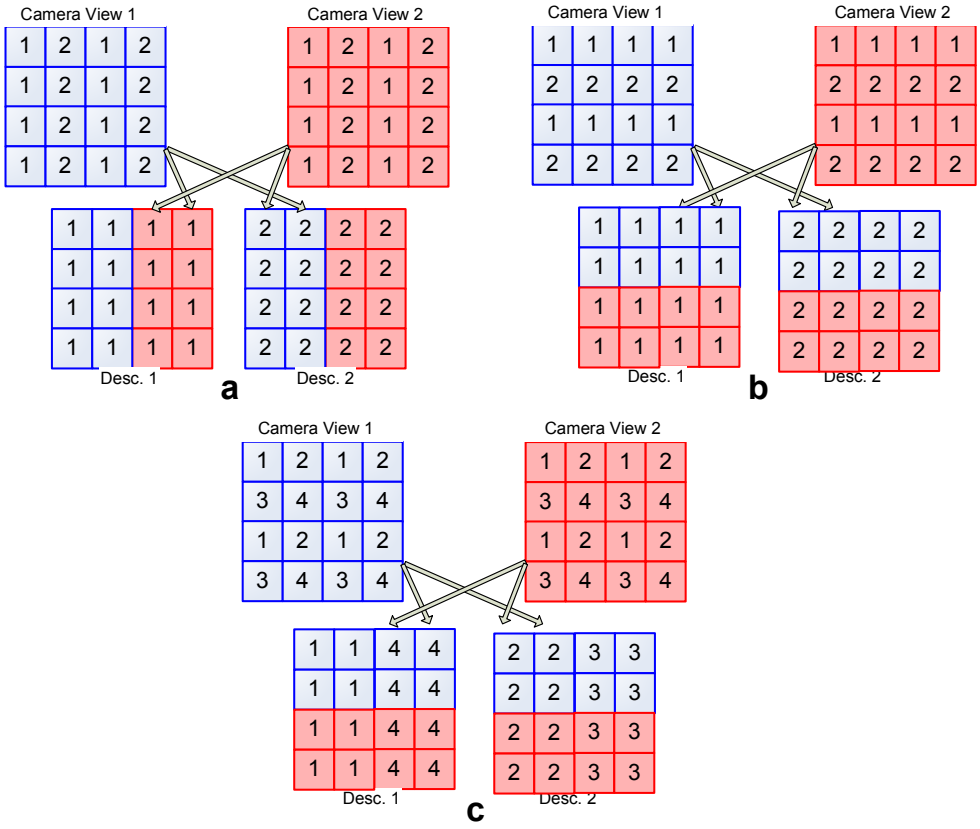

Figure 34 - Different tested multi-description generation methods

The first two schemes in Figure 34, i.e., a and b, split the two camera views by combining their alternating columns and rows, respectively. As a result, two descriptions are produced for a pair of camera views, both of which are identical in spatial resolution. The third scheme (c) refers to quincunx sampling of two camera views and combining the samples in two different descriptions, similar as in a, and b. Following the transmission of descriptions through two separate links (i.e., these correspond to different P2P sub-trees), descriptions are decoded individually and independently with SVC decoders, and the decoded descriptions are merged together to reconstruct the stereoscopic camera views (Camera View 1 and Camera View 2 in Figure 34).

For the tests, two of the ROMEO captured contents are used: Panel Discussion and Martial arts. Readers can refer to Deliverable D3.1 - Report on 3D media capture and content preparation to learn more information about both contents. Panel discussion is a fairly static scene with sitting panel members with minimal movement. Thus, temporal activity in the scene is very limited throughout the scene. On the other hand, Martial Arts scene is a highly active scene with two kick-box fighters and the referee. The temporal activity is significantly higher than in Panel Discussion scenario. It is wanted to see the efficiency of different MDC schemes on different type of contents.

Following the formation of descriptions according to the three schemes depicted in Figure 34, each description has been encoded with SVC (using JSVM encoder) to produce a single Medium Grain Scalability (MGS) quality enhancement layer. In order to make a fair comparison, in each MDC scheme, each description has been encoded to generate equal base layer and enhancement layer bit-rate. The bit-rates are depicted in Table 6.

	Base layer	Base + Enhancement Layer
Panel Discussion	3300 Kbps	7400 Kbps
Martial Arts	6500 Kbps	13600 Kbps

Table 6 - Stereoscopic video stream bit-rates (total bit-rate of both descriptions)

To make rate-distortion efficiency comparison with single description scalable stereoscopic video streaming, both camera views have also been encoded individually with two quality layers and at the same bit-rates depicted in Table 6. Both description streams have been streamed from the server to the client, assuming different random packet loss patterns of 3% and 10%. Both paths have been assumed to have similar conditions in each test. Due to the lack of a P2P multi-tree delivery simulator for multi-description coded SVC streams at the time, initial investigation has been performed using a random packet erasure model. No priority rule has been applied to base layer slices over enhancement layer slices in both paths. This means that, when the base layer slice is hit, corresponding enhancement layer slice is forced to be discarded. In the case of single description coding, Camera View 1 and Camera View 2 streams have been transmitted across two separate paths along with their quality enhancement layers. After decoding the descriptions and merging them, the reconstructed L-R stereoscopic video quality is calculated using the perceptual QoE estimation model developed in WP3 of ROMEO project. The quality index varies in [0, 1] interval. 0 means perfect quality (identical with the reference) and 1 means the worst quality. The results are depicted in Table 7 and Table 8 for Panel Discussion and Martial Arts, respectively.

Description Generation method	No packet losses			3% packet loss rate			10% packet loss rate		
	highest	medium	lowest	highest	medium	lowest	highest	medium	lowest
method a	0.126	0.157	0.207	0.128	0.160	0.210	0.134	0.169	0.219
method b	0.124	0.153	0.202	0.126	0.157	0.206	0.133	0.167	0.218
method c	0.225	0.346	0.501	0.227	0.349	0.503	0.229	0.355	0.506
single desc.	0.101	0.110	0.121	0.109	0.118	0.131	0.122	0.149	0.185

Table 7 - Results for Panel Discussion

Description Generation method	No packet losses			3% packet loss rate			10% packet loss rate		
	highest	medium	lowest	highest	medium	lowest	highest	medium	lowest
method a	0.148	0.218	0.326	0.149	0.221	0.328	0.154	0.231	0.335
method b	0.141	0.201	0.300	0.142	0.203	0.302	0.148	0.216	0.312
method c	0.325	0.474	0.626	0.326	0.477	0.627	0.329	0.487	0.628
single desc.	0.090	0.109	0.137	0.168	0.299	0.382	0.391	0.429	0.474

Table 8 - Results for Martial Arts

“Highest” refers to the case, where the enhancement layers of both descriptions are streamed from the server. “Medium” refers to the case, where the enhancement layers of one of the descriptions is dropped and “lowest” refers to the case, where the enhancement layer streams of both descriptions are dropped. The results suggest that the performance of the multiple-description generation scheme is strongly affected by the type of the content. For stationary scenes, where the temporal activity is very limited, spatial error concealment methods offer

perceptually good results in single description coding, thus penalising the reduced description quality in multiple descriptions. On the other hand, for temporally active scenes, error concealment methods fail to recover the perceptually detrimental effects of packet losses in the network. Thus, multiple description methods, especially methods a and b (as indicated in Figure 34) perform perceptually significantly better. This shows the importance of the applied multi-phase subsampling based description generation scheme. Since the multiple description generation method c (quincunx) sampling results in noticeably increased amount of spatial frequencies in each description frame, the increase in the overall bit-rate results in the dropped reconstruction performance. On the other hand, methods a and b show very similar performance in terms of perceptual reconstruction quality, while method b (top-bottom) is slightly better.

The aforementioned multiple description methods will be tested in a next step with asymmetric quality coding as well as region of interest based encoding (depending on the delivered visual attention models) to test the perceptual stereoscopic video reconstruction performance. Furthermore, following the release of the initial network model that simulates the P2P multi-tree multicast delivery for SVC coded descriptions, more realistic performance indications will become available.

4.2.2 Experimental results of rate adaptation in parallel with MDC

As described previously, an experimental test-bed has been implemented to obtain results from streaming SVC side by side video over MPEG2-TS/P2P Chunk/UDP network. The resulted PSNR measurements in Figures 27-31 indicate that the application of MAP improves the quality of the received SVC side-by side video significantly as opposed to not performing any transmission rate adaptation scheme. In this section, PSNR measurements of the left-right video obtained from the proposed MDC decoding scheme, are presented. . As per one of the objectives of ROMEO project, which is the guaranteed minimum Quality of Service to all users, DVB and P2P are combined for multi-description delivery. The other side-by-side description of the stereoscopic view is compressed and broadcast over DVB to the client.

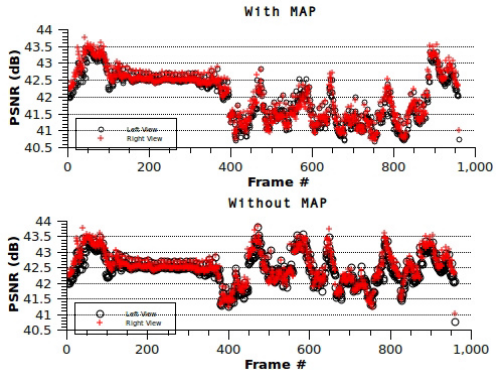

Figure 35 - Comparison of PSNR per frame of MDC left-right sequence obtained with and without enabling MAP (6Mbps B/W)

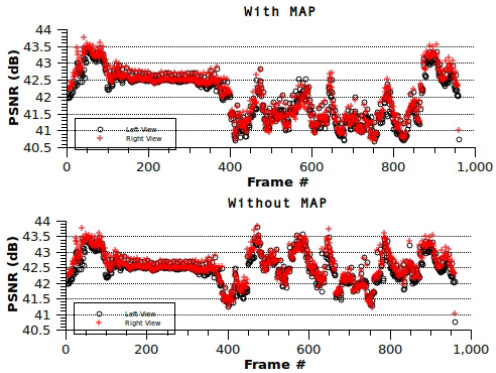

Figure 36 - Comparison of PSNR per frame of MDC left-right sequence obtained with and without enabling MAP (6.25Mbps B/W)

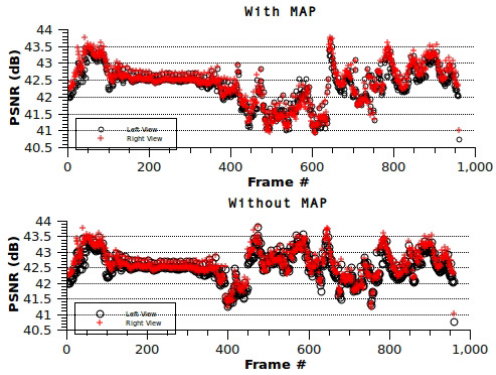

Figure 37 - Comparison of PSNR per frame of MDC left-right sequence obtained with and without enabling MAP (6.5Mbps B/W)

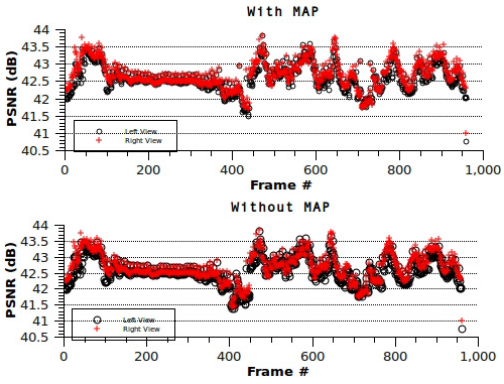

Figure 39 - Comparison of PSNR per frame of MDC left-right sequence obtained with and without enabling MAP (7Mbps B/W)

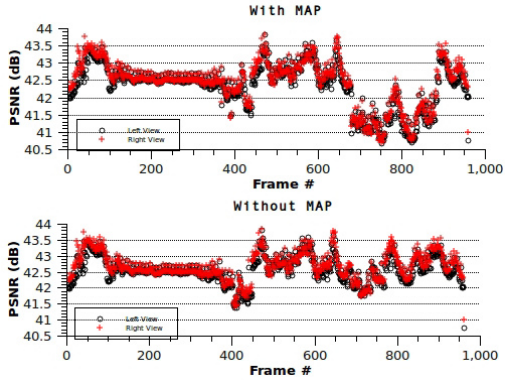

Figure 38 - Comparison of PSNR per frame of MDC left-right sequence obtained with and without enabling MAP (6.75Mbps B/W)

Figures 35-39 illustrate the comparison of PSNR per frame obtained when the MAP adaptation is enabled or not, for left and right views under different network conditions. It can be seen that MDC error concealment succeeds on improving the quality of the received video significantly in almost all cases, compared to the PSNR obtained by SVC side-by-side decoder, as in Figures 27-31. Moreover, it has to be mentioned that there is only marginal decrease in the PSNR when the MAP transmission rate adaptation is applied. The major reason for this is that the DVB stream (almost 3Mbps) is received without losses. Hence, lost and non-reconstructed macroblocks of the decoded P2P video frames are replaced with the pixel blocks interpolated from the decoded DVB video frames.

Available Bandwidth (Mbps)	With MAP		Without MAP	
	Left view	Right View	Left view	Right View
6	41,99 dB	42,06 dB	42,44 dB	42,52 dB
6.25	42,02 dB	42,09 dB	42,47 dB	42,56 dB
6.5	42,33 dB	42,41 dB	42,53 dB	42,62 dB
6.75	42,34 dB	42,43 dB	42,57 dB	42,67 dB
7	42,59 dB	42,69 dB	42,62 dB	42,71 dB

Table 9 - Average PSNR obtained from the different experimental scenarios

As shown in Table 9, the MDC error concealment achieves an improvement of 5dB to 9dB in PSNR, with MAP and without MAP respectively, compared to the measurements in Table 5. This is due to the fact that, when the received descriptions are merged in a post-decoder stage, they are blended equally. The exception to this however is that that when P2P slices are missing due to packet losses and not reconstructed properly during decoding, those faulty pixels are interpolated using DVB reconstructed values. When the MAP is not used, a smaller number of enhancement layer packets are dropped, which results in more successfully reconstructed frames of higher quality. Table 10 illustrates the Packet Loss Rate for both layers when MAP is used and when it is not.

Available Bandwidth (Mbps)	With MAP		Without MAP	
	Base Layer PLR (%)	Enhancement Layer PLR (%)	Base Layer PLR (%)	Enhancement Layer PLR (%)
6	5.95	6.35	0.21	59.68
6.25	3.79	4.29	0.24	57.45
6.5	2.35	2.23	0.31	26.03
6.75	1.25	1.17	0.41	25.70
7	0.48	0.47	0.51	0.43

Table 10 - Average PLR obtained from the different experimental scenarios

In addition, the lost base layer slices, which cannot be successfully reconstructed, are not reflected in the resultant left and right view frames. On the other hand, when MAP is used, a lot of the reconstructed video frames are at lower quality, due to the fact the MAP primarily selects to drop the entire enhancement layer as long as the transmission rate exceeds the bandwidth threshold. This results in marginally worse PSNR after the MDC decoding, despite the fact that the number of lost base layer slices is substantially smaller in the latter case, and hence the achieved continuity is better.

The near optimum video quality resulted from applying MDC error concealment relies on the assumption that a guaranteed DVB stream (frame compatible side-by-side) next to the internet stream that leads to Full HD 3D-TV, is always available. This assumption is valid, as it is defined as a sub-set of the ROMEO use cases.

Nevertheless, MAP transmission rate adaptation as described in section 4.1 will be necessary under real time streaming over wireless channels, when available bandwidth is scarce and MAC layer retransmissions quickly consume available resources. The MAP will significantly improve the 3D video delivery performance for portable laptop users, by selectively discarding the descriptions (color plus depth) of the side cameras and even depth maps of the center camera pair (as described in D4.1), for which no substitute description arrives through the DVB channel. This study is currently undergoing.

4.3 Fail-over mechanism at P2P level

Among other functionalities, the chunk selection module incorporates a fail-over mechanism. The fail-over mechanism is designed to mitigate the effects of churn. Due to churn, peers get disconnected for short or long periods. After a peer or a group of peers leave the network, some peers get disconnected. Disconnection periods are related to the available service capacity. By the term ‘service capacity’ we refer to the capacity of the network in terms of peers that can be served by other peers. It is a metric related to the application layer rather than the network layer. Normally, different sub-trees differ in terms of available capacity. The only way to homogenize the available capacity is using the so-called *peer migration*.

According to peer migration, peers change properties in each sub-tree, i.e. they change their status from sterile to fertile and reversely. However, migration itself causes additional disconnections and induces increased computational overhead for the super peer.

If the available service capacity in a sub-tree is negative, some peers will be disconnected in this tree for possibly long periods. This is because only the arrival of a fertile peer can increase the service capacity. Not all of the disconnected peers will have the opportunity to disconnect until the service capacity becomes non-negative. Short-term disconnections can occur if the service capacity remains non-negative after the effect that caused change in the structure of the tree.

The fail-over mechanism of the chunk selection module is based on the facts that: a) a peer can remain connected in some trees after a disconnection event and b) the parent peers in these non-disconnected trees can possibly possess chunks that belong to the flows traversing the sub-trees in which the peer is disconnected. Therefore, the disconnected peers can request important chunks from its connected parents.

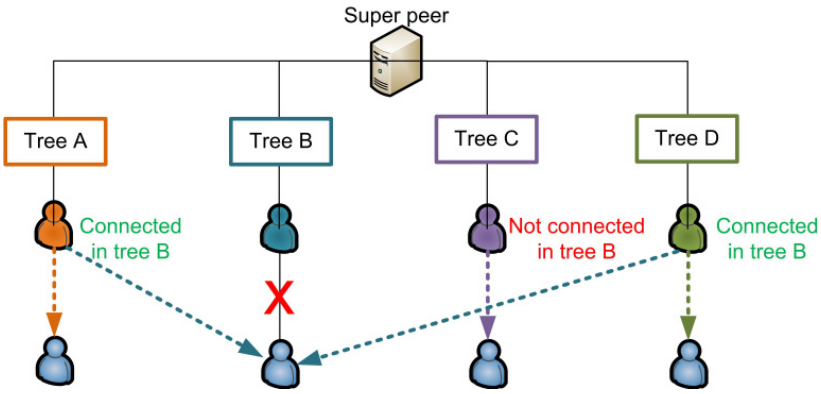

Figure 40 - Fail-over via chunk selection

In Figure 40, a peer is disconnected in tree B. However, it remains connected in trees A, C and D. In addition, its parent peers in trees A and D are connected in tree B. These peers can provide the disconnected peer with important chunks delivered through tree B.

The chunk selection module resides inside the peer client. Missing chunks are requested and forwarded according to their priority, which is extracted from the corresponding field of the chunk header.

5 CONCLUSIONS

In this document, detailed specifications concerning packetization issues in ROMEO are provided. Novel packetization schemes are proposed in order to optimize the transport of 3D audio channels and multiview video content over the targeted hybrid network environment (DVB and P2P).

As well with the packetization and media partitioning, additional metadata and signalling mechanisms (DVB) are specified in order to allow synchronizing the different streams and to ease the rendering at client side.

Finally, adaptation and error resilience mechanisms are proposed and validated with preliminary experimental results.

6 REFERENCES

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

[2] D. Hoffman, G. Fernando, "RTP Payload Format for MPEG1/MPEG2 Video", IETF RFC 2250, January 1998.

[3] ISO/IEC 14496-10:2009 International Standard, “Information technology — Advanced video coding for generic audiovisual services, March 2009

[4] Y.-K. Wang, R. Even, T. Kristensen, R. Jesup, “RTP Payload Format for H.264 Video”, IETF RFC 6184, May 2011

[5] S. Wenger, Y.-K. Wang, T. Schierl, A. Eleftheriadis, “RTP Payload Format for Scalable Video Coding”, IETF RFC6190, May 2011

[6] Y.-K. Wang, T. Schierl, R. Skupin, P. Yue, “RTP Payload Format for MVC Video”, IETF draft-ietf-payload-rtp-mvc-02, June 25, 2012

[7] ISO/IEC 13818-1:2007 International Standard, “Information technology — Generic coding of moving pictures and associated audio information: Systems”, October 2007

[8] ETSI EN 302 755 V1.2.1 European Standard: Digital Video Broadcasting (DVB); Frame structure channel coding and modulation for a second generation digital terrestrial television broadcasting system (DVB-T2); 2011.

[9] Alex MacAulay, Boris Felts, Yuval Fisher, White Paper – IP Streaming of MPEG4: Native RTP vs MPEG-2 Transport Stream, October 2005