Project Number: 248495
Project acronym: OptiBand
Project title: Optimization of Bandwidth for IPTV Video Streaming

Deliverable reference number: D4.5.2
Deliverable title: Head end prototype Version 2 and associated documentation

Due date of deliverable: M22
Actual submission date: 30 November 2011 (M23)

Start date of project: 1 January 2010 Duration: 30 months

Organisation name of lead contractor for this deliverable: Fraunhofer HHI
Name of the lead author for this deliverable: Yago Sánchez

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination Level PU		
PU	Public	x
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n 248495

Contributors

Participant #	Participant short name	Name of the Contributor	E-mail
1	HHI	Yago Sánchez	yago.sanchez@hhi.fraunhofer.de
2	HHI	Karsten Grüneberg	karsten.grueneberg@hhi.fraunhofer.de
3	TVN	Dumas Olivier	olivier.dumas@thomson-networks.com
4	LAMBDA	Samuel Rivas	samuel.rivas@lambdastream.com
5	IRD	Dmitri Jarnikov	djarnikov@irdeto.com
6	UDC	Miguel Barreiro	miguel.barreiro@udc.es
7	OPTEC	Alex Chernilov	alexc@optibase.com
8	CSL	Yossi Barsheshet	yossibs@corrigent.com

Table of Contents

Table of figures

Table of Tables

Glossary

Abbreviation / acronym	Description
AAC	Advanced Audio Coding
AAC-LC	Low complexity Advanced Audio Coding
ADSL	Asymmetric Digital Subscriber Line
ARM	A 32-bit reduced instruction set computer instruction set architecture developed by ARM Holdings
ASI	Asynchronous Serial Interface
ATM	Asynchronous Transfer Mode
AVC	Advanced Video Coding
BSS	Business Support Systems
CA	Conditional Access
CAS	Conditional Access System
CAT	Conditional Access Table
CBR	Constant Bit Rate
CC	Continuity Check
CW	Control word
DMA	Direct memory access
DMZ	De-militarized zone - an internal protected network that is outside the firewall (also known as a peripheral network)
DSL	Digital Subscriber Line
DSLAM	Digital Subscriber Line Access Multiplexer
DTS	Decode Time Stamp
DTV	Digital Television
DVB	Digital Video Broadcasting (forum)
ECM	Entitlement Control Messages carry the keys that are used to unlock the encrypted content
EPG	Electronic program guide
ES	Elementary stream
FIFO	First In, First Out
GOP	Group Of Picture
GUI	Graphical user interface
H.264/MPEG-4 AVC	Codec developed by the ITU-T Video Coding Experts Group
HD	High Definition
HDMI	High-Definition Multimedia Interface
HE-AAC	High-Efficiency Advanced Audio Coding
HTTP	Hypertext Transfer Protocol

IANA	Internet Assigned Numbers Authority
IDR	I-frame known as an instantaneous decoder refresh (IDR) frame
IEC	International Electrotechnical Commission
IP	Internet Protocol
IPTV	Internet Protocol television
ISO	International Organization for Standardization
JSVM	Joint Scalable Video Model
LAN	Local area network
LAN RX	Local area network “receiver”
LAN TX	Local area network “transmitter”
LUT	Look-up table
Metro Network	A metropolitan area network (MAN) is a network that usually spans a city. An MAN usually interconnects a number of local area networks (LANs) using a high-capacity backbone technology, and provides uplink services to wide area networks (or WAN) and the Internet.
MGS	Medium Grain Scalability
MIPS	A reduced instruction set computer instruction set architecture developed by MIPS Technologies
MOS	Mean opinion score
MPEG	Moving Picture Experts Group
MPEG-2 TS	Transport stream format specified in MPEG-2 Part 1, Systems (ISO/IEC standard 13818-1).
MZ	Militarized zone - an internal protected network behind the firewall
NAL	Network Abstraction Layer
NIT	Network Information Table
nPVR	Network Personal Video Recorder
OP	Operation Point is defined as a unique combination of temporal and quality levels
PAT	Program Association Table
PCR	Program Clock Reference
PDA	Packet Dropping Algorithm
PDD	Packet Dropping Device
PID	Packet Identifier
PMT	Program Map Table
PPS	Picture Parameter Set
Priority ID	Priority ID is associated to the pair of {T (temporal layer), Q (quality layer)} provided by the NAL unit header.
PSI	Program Specific Information, metadata about a program (channel) and part of a MPEG transport stream, the PSI data contains five tables: PAT (Program Association Table)

	CAT (Conditional Access Table) PMT (Program Map Table) NIT (Network Information Table) TDT (Time and Date Table)
PSNR	Peak signal-to-noise ratio
PT	Payload type
PTS	Presentation Time Stamp
PVR	Personal video recorder
Quality ID	The value of the quality_level syntax element that is present in the scalable extension NAL unit defined in the SVC video specification. This non-negative integer indicates the quality level that the sample would provide.
RAP	Random access points
RTP	Real-time Transport Protocol
RTSP	Real Time Streaming Protocol
SCR	System Clock Reference
SD	Standard definition
SDI	Serial digital interface
SDL	Simple DirectMedia Layer
SNMP	Simple Network Management Protocol
SNR	Signal-to-noise ratio
SPS	Sequence Parameter Set
SPTS	Single Program Transport Stream
SSRC	Synchronization source
STB	Set-Top Box
SVC	Scalable Video Coding
TDT	Time and Date Table
Temporal ID	Temporal ID
TS	Transport Stream
T-STD	Transport Stream system target decoder
UDP	User Datagram Protocol
URL	Uniform Resource Locator
VBR	Variable Bit Rate
VLC	media player and multimedia framework written by the VideoLAN project
XML	Extensible Markup Language
YUV	Colour space presentation

1. Executive Summary

The main goal of this project is to develop an efficient solution for IPTV operators to provide HD video streams to users which have a limited access bandwidth and to optimize bandwidth usage for IPTV video streaming. WP4 deals with the Head End solution.

This document describes the results of Task 4.5 “Build and test prototypes” of the OptiBand project. This deliverable includes the description of the final version of the Head End prototypes. In the following the two different approaches developed in OptiBand are described:

Method 1: AVC video encoding with multiple profiles

In this method, multiple streams per channel are created at the Head End by an H.264/AVC encoder and are encapsulated in MPEG-2 TS, having each stream a different bit rate. The streams belonging to the same channel are synchronized to allow the packet dropping device (PDD) to seamlessly switch from one stream to another. The PDD receives the multiples streams per channel generated at the Head End but outputs only one stream per channel, dynamically switching between the streams based on the conditions of the ADSL link and the combination of competing channels over the same shared ADSL link.

Method 2: SVC video encoding

In this method, a single stream per channel is created at the Head End by an SVC encoder using MGS SNR scalability [9][12]. The stream is encapsulated in MPEG-2 TS packets and, subsequently, in a set of RTP packets. This method allows dropping data that carries SNR (i.e. quality) enhancements to a picture. In the following, it is referred to *quality_id* as quality level, which indicates the MGS layer a NAL unit belongs to, and to *temporal_id* as temporal level, which indicates importance of a NAL unit in the temporal domain, being a NAL unit with temporal level *i* dependent on NAL units with temporal level 0...*i*-1. Based on the quality levels and temporal levels, different operation points are defined. A certain operation point OP_x is associated to a tuple $\{T_x,Q_x\}$, where T_x and Q_x refer to a certain temporal level and quality level respectively, and entails the set of all NAL units with a quality level and a temporal level lower than or equal to the ones in the associated tuple. Since the temporal level and quality level are signalled in the NAL unit header and this data is usually not accessible due to the data being encrypted, for this method the SSRC field of the RTP packets are used for signalling the necessary information so that the PDD can associate the incoming data to the appropriate operation point selected for output. The bit rate reduction is achieved by dropping packets that contain data outside of the chosen operation point.

2. Introduction

The deliverable D.4.5.2 – Head End prototype Version 1 and associated documentation describes the stable version of the Head End Prototypes of two of the approaches developed in OptiBand for packet dropping: AVC video encoding with Multi profiles and SVC video encoding with MGS scalability. This document builds the technical basis of all technologies involved in the Head End. It includes the Head End prototype description, encryption components, middleware, STB, configuration of the network, interaction between Head End and Packet Dropping Device, used metadata information as well as Head End prototype conformance check and system test.

3. Head End Prototypes

3.1 Head End solutions

3.1.1 AVC video encoding with multiple profiles

In the multiple profiles approach, encoders at the Head End generate multiple copies of each video content, with each copy having a different bit rate and hence different quality.

In this case, the multiple qualities generated at the Head End are distributed through the network to allow the Packet-Dropping Device (PDD) to choose which bit rate stream is the most suitable for each of the requesting clients. The decision made at the PDD is based on the client's available access bit rate and the cumulative bit rate of other channels that compete on the same user access links.

Two multiple profiles approaches are presented for Head End prototype: the first one is called H264/AVC Single video SPTS with external multiplexer and the second one H264/AVC Multi-video SPTS with embedded multiplexer.

3.1.2 SVC video encoding

In the SVC-based approach, the Head End generates a single SVC stream per content, which is distributed to the network and the PDD adapts the stream sending an SVC stream as output only with the number of layers that match the available throughput of the access links of the clients. For this purpose, in order to allow switching from one layer to another at the PDD at any point of time, MGS SNR scalability [9] is considered.

Using MGS allows having several Operation Points (OPs) within a single SVC encoded bit stream. The different OPs are valid SVC sub-streams, which correspond to different bit rates. Therefore, for the SVC approach, a single video stream is needed per video content, reducing the load at the Head End and the bandwidth consumption at the metro network (between the Head End and the PDD) in comparison to the AVC-based approach, where multiple copies of the same content are sent at different bit rates.

Obviously, this kind of video adaptation can be implemented in a very efficient way. The adaptation is performed by a proxy application that runs on the network device (PDD) and adapts the video stream on-the-fly as it is transmitted over the network. The SVC-based adaptation just requires dropping of selected IP packets [10]. An interesting approach which improves IPTV services by SVC Adaptation is presented in [11]. All this in addition to the fact that SVC capable STBs are currently being developed, see [14], makes the SVC-based approach very interesting for future IPTV services.

The SVC bit rate adaptation approach is based on multiple Operation Points (OPs) within the same MGS-encoded scalable stream. A certain operation point OP_x is associated to a tuple $\{T_x,Q_x\}$, where T_x and Q_x refer to a certain temporal level and quality level respectively, and entails the set of all NAL units with a quality level and a temporal level lower than or equal to the ones in the associated tuple. The OPs are defined in such a way a higher OP is a superset of the OP directly below in order to allow simple successive dropping of data to obtain the different OPs.

Each OP refers to a valuable stream at a certain quality level with a corresponding bit rate. In the Head End, the data units are tagged with priority levels. When passing the PDD, the data rate of the stream is monitored and compared with the data rate available on the DSL link to the SVC client. If the incoming data rate exceeds the available data rate, the PDD drops part of the data stream forwarding a lower OP. Dropping starts with NAL units belonging to the highest OP. As aforementioned the signalling mechanism used is based on the indication at the SSRC field of the RTP packet where a value of Priority ID is inserted. In Table 1 an example of how to assign the OPs to the NAL units and the assignation of Priority ID is shown. The table also shows the achievable rates at each OP for an SVC stream encoded with a GOP 8 of a 10s long sequence with 1080p at 25 fps. The Priority IDs are assigned in such a way that dropping starts with NAL units belonging to higher highest Priority ID value. All temporal levels of the base layer are always forwarded, i.e. the lowest OP is the complete base layer, no temporal scalability is considered for base layer with this approach. This resulting number of OPs in this example is 9 (tagged with Priority ID 0..8).

Table 1 – Priority assignment to each OP (T, Q layer combination)

OP	T:	0			1			2			3			Rate	PSNR
	Q:	0	1	2	0	1	2	0	1	2	0	1	2	[kbps]	[dB]
1		●	●	●	●	●	●	●	●	●	●	●	●	8225,4	40,841
2		●	●	●	●	●	●	●	●	●	●	●	x	7121,6	39,847
3		●	●	●	●	●	●	●	●	X	●	●	x	6295,9	39,211
4		●	●	●	●	●	X	●	●	X	●	●	x	5703,1	38,836
5		●	●	x	●	●	X	●	●	X	●	●	x	4559,6	38,506
6		●	●	x	●	●	X	●	●	X	●	x	x	3960,2	37,764
7		●	●	x	●	●	X	●	x	X	●	x	x	3521,5	37,201
8		●	●	x	●	x	X	●	x	X	●	x	x	3220,2	36,835
9		●	X	x	●	x	X	●	x	X	●	x	x	2670,7	36,437
PriorityID		0	1	5	0	2	6	0	3	7	0	4	8		

The SVC-based Head End provides H.264/SVC bit streams to the SVC-based Packet Dropping Device. These bit streams are encapsulated into MPEG-2 TS [17] and additionally into RTP [20] packets. For this purpose, the MPEG-2 TS Encapsulator and RTP Encapsulator at the Head End for SVC-based approach have been developed.

Only TS packets that contain NAL units from the same quality level and temporal level are sent within the same RTP packet. Packets that belong to different temporal levels are encapsulated in separate RTP packets. By these means, the Packet Dropping Device can either forward or drop a complete RTP packet. Figure 1 illustrates the functionality of the encapsulation and the aforementioned additional tagging process of SVC data using RTP header. More details about the SVC encapsulation approach for MPEG-2 TS is found in [4] with corresponding diagrams (Figures 22-24).

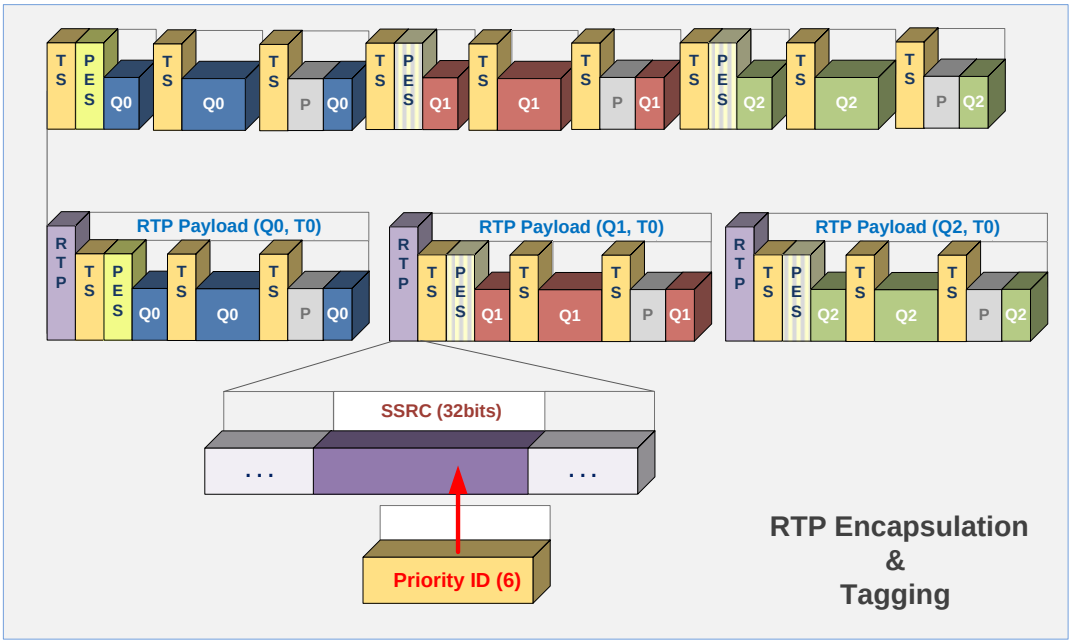


Figure 1 – RTP/MPEG-2 TS encapsulation of SVC & Tagging with Priority ID

3.2 Conditional access and encryption

IPTV services use encryption to protect a service from unauthorized viewing and manipulation. The process of encryption involves the use of a mathematical procedure, or algorithm, which operates on the digital data being protected, in conjunction with a secret piece of information called a 'control word'. The operation

produces encrypted data, which is meaningless to anyone except those who possess the decryption control word to recover the correct data. Smart cards and/or secure software modules are able to recover the control word in a secure manner. The control word is sent to the customer in an encrypted Entitlement Control Message (ECM). Control words change on a regular basis and each ECM always contains two control words (current and next) to ensure continuous viewing. ECMs are specific to the scrambled content and are the same for all subscribers.

The scrambler is a software library that creates control words, generates ECMs, and uses the control words to scramble the data, and stores the ECMs together with the scrambled data. The scrambler library performs scrambling at the MPEG-2 transport level. ECMs are inserted in place of the existing null packets in the MPEG-2 TS. Most transmission schemes impose strict constant bit rate requirements on the transport stream, and if the transport stream bit rate is higher than the cumulative bit rate of all content streams that it carries, a multiplexer may need to insert some additional packets (called null packets) to maintain the bit rate when there are no sufficient TS packets to be sent. The payload of null packets may not contain any useful data at all, and the receiver is expected to ignore its contents. The multiplexer that precedes the scrambler must be configured in a way that guarantees a minimal amount of null packets in a transport stream. For the Head End prototype scrambler needs at least one null packet per 300 msec.

The scrambler application is a software component, built on top of the scrambler library that is designed to receive streams of live IP data, scramble them with a control word and transmit them to the user equipment.

3.3 IPTV Middleware

The IPTV middleware is a client-server software that connects end users (through the middleware client running in their STBs) to the video Head End. It allows end users to browse media content available in the network, watch videos from the streaming servers, and manage their own preferences such as GUI language, video language, etc. On the server side, the middleware controls user access rights (so that users only have access to what they have paid for), media acquisition (registering or deleting content from the IPTV platform) and the miscellaneous services that could integrate in an IPTV platform like server-side PVR, EPG, teletext, etc.

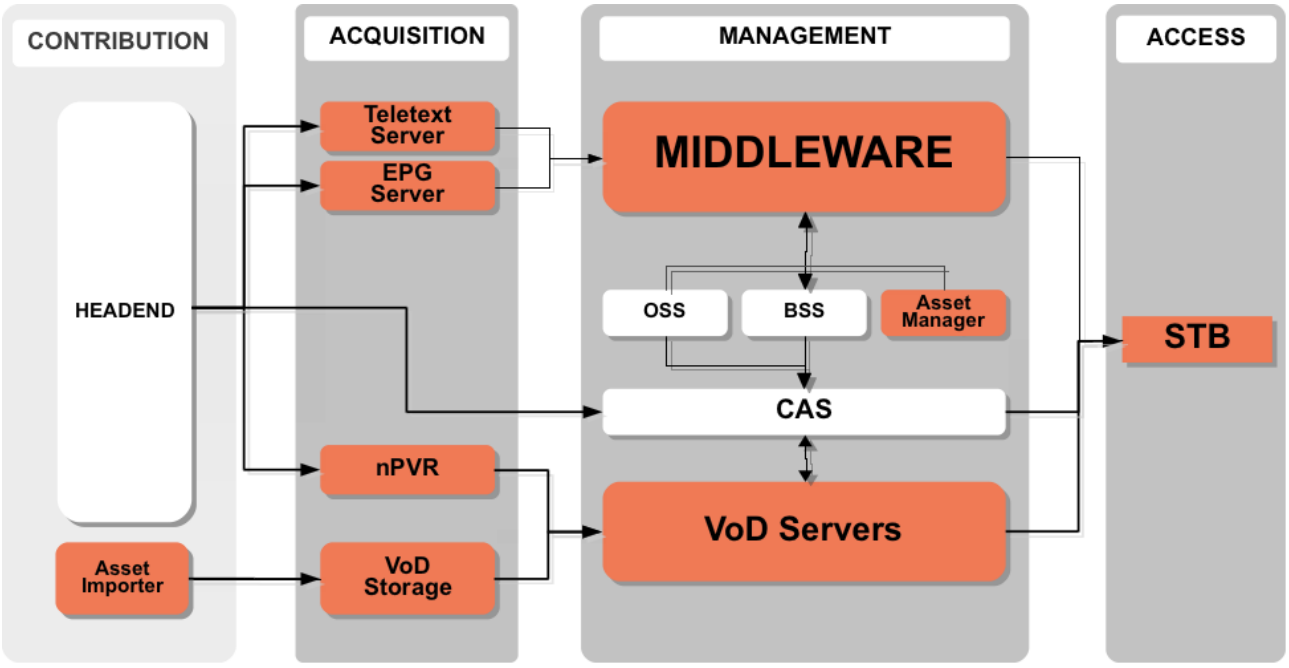


Figure 2 – Middleware deployment

- Content providers must introduce the media metadata in the middleware in order to make it available in the service. If the metadata is already in a supported format, the process is automatic (using the asset importer), otherwise it should be uploaded manually to the asset manager. The prototype implementation uses strictly manual uploading of metadata. This approach does not require a significant workload since the amount of assets that would be available is very small. In real

scenarios, the content provider and the middleware vendor must integrate to implement an automatic metadata uploading process.

- The EPG server loads metadata for broadcast media automatically as long as it is signalled in the transport stream using DVB PSI tables. In the prototype testing no broadcast streaming will be used.
- Similar to the EPG, the teletext server reads teletext information from the transport stream PSI tables. As for EPG, in the prototype testing no teletext will be used.
- VoD Servers and VoD Storage. Almost any video server will use standard protocols such as HTTP, transport stream over UDP or transport stream over RTP (with RTSP as session control protocol). The integration with the middleware simply requires that an URL is associated with video metadata so that the STB can request the appropriate stream from the VoD server when the user selects a video for watching. Even though both RTSP and HTTP have registered official URL formats in the IANA, transport stream over UDP have not. LambdaStream's middleware uses the format `udp://<multicast-address>:port` for that purposes. This doesn't affect to the video servers, but it must be taken into account when generating the metadata to connect video information to the actual stream source.
- The nPVR server records broadcast data so that it is available as VoD content. This will not be considered for the tests.
- The BSS controls the rights and purchases of the users, providing support for different business models such as subscription, pay per view, renting, pay per use, etc. The middleware uses the BSS to notify the purchases, rentings, etc that the user does through the GUI in the client middleware and to query the rights users have in order to represent them in the GUI (for example presenting lists of videos to purchase, lists of purchased videos or even hiding videos that users are not allowed to watch). The BSS integrates with a CAS (external to the middleware) in order to protect content according to user rights.

There are two new optional interfaces that will contribute to get better performance of the PDD, leading to the better user experience:

- An interface between the client middleware and the PDD's user preference module, so that end users can configure the priorities of STBs in their households, and the priorities for the different types of media. For example, users will probably set higher quality for the STB connected to the larger TV in their living room than for the smaller TV in the kids' room, or more priority to sports than to news.
- An interface from the PDD to the middleware server, so that the PDD can query the metadata corresponding to the media being streamed to each user. This is needed to implement the possibility of giving different priorities to different types of media.

The physical distribution of the middleware spans across several layers in the overall architecture of an IPTV platform. The middleware client is deployed in the user home networks. In the case of OptiBand, this network will be connected to a DSLAM that connects several homes to the carrier network. In the server side, the middleware components spread in the DMZ, the application layer and the MZ for security and reliability reasons. For simplicity, in this prototype the whole middleware will be deployed in a single server.

4. Multi bitrate based Head End prototype

4.1 Introduction

This chapter introduces both hardware and software elements used of the Head End prototype for the two multi-bitrate methods:

a) N x [H.264/AVC Single video SPTS] with Multiplexer, where N is the number of Single Video SPTS that will be multiplexed into a Multi video SPTS.

b) H.264/AVC Multi-video SPTS with embedded Multiplexer.

This description includes values of the configured parameters, the metadata information that the Head End produces and the video streams structure that the prototype generates. Furthermore, this deliverable reports which streams are created for testing the performance of the PDD in [6]. For the second version of the prototype, the goal is to achieve 33% bandwidth reduction. For this purpose, up to four streams may be generated per video content at the Head End (one of them with a bit rate 33% lower than the highest bit rate), allowing the PDD for adapting the forwarded streams at different levels depending on the scenario. The streams generated at the Head End are assigned to different *bandwidth_levels* (bw_level x) and each *bandwidth_level* has assigned a given MOS value, which allows the PDD to prioritize streams based on the expected subjective quality indicated by the assigned MOS values. Table 2 shows the look-up-table containing the MOS values of the different stream content types that are used in the PDD for the selected bit rates.

The MOS values in Table 2 are extracted from the results shown in [7]. However, since in [7] only subjective test for Action movies und Soccer movies were carried out, the MOS values for the documentary have been calculated as the average of both of them as shown in the table.

Table 2 - Look-up-Table (LUT): MOS values for the bandwidth_levels for the multirate approaches

Stream_type	bw_level 0	bw_level 1	bw_level 2	bw_level 3
Action	4.30	4.14	4.07	3.88
Soccer	4.10	4.02	3.98	3.81
Documentary	4.20	4.08	4.025	3.845

Based on the evaluation on QoE done in [7], a set of bit rates for the different streams is selected to provide the possibility of applying an efficient algorithm at the Packet Dropping Device (PDD). As concluded in [3], ideally a very high amount of streams at different bit rates would give a higher freedom to the PDD to forward one or another stream version depending on the DSL link sharing scenario. However, since there is a limitation on the capabilities of the encoders in terms of the amount of version that can be generated of a given content, only four different versions will be generated at most. It is important to distinguish between the bit rates at the application layer (MPEG-2 TS) and the bit rates at the physical layer, which is at the ADSL layer. The bit rates at the application layer are referred to as the multiplex, that is, they are the sum of video, audio, PSI tables and stuffing. The bit rates for the streams (at the application layer) that will be generated are: 7.76, 6.31, 4.80 and 3.35 Mbps that would result in about 8.00, 6.50, 4.95 and 3.45 Mbps after encapsulation for transport (i.e. UDP/IP/Ethernet). The MOS values for the selected rates obtained from subjective tests (see [7]) are summarized in Table 2, where the *bandwidth_levels* are ordered from the highest bit rate to the lowest bit rate, i.e. bw_level_0 corresponds to the highest rate (i.e. 7.76 Mbps) and bw_level_3 to the lowest one (i.e. 3.35 Mbps).

The selected rates defined based on the conclusion of the research of [7], the ADSL link constraints defined at [1], and of course from the objectives of the project as defined at [21]. All rates qualities have to be acceptable based on the subjective tests performed at [7] and to meet the objective quality as defined at [21]. The ADSL profiles defined at [1] are 15mbps and 10mbps and the selected rates achieve reasonable statistical multiplexing in order to meet the 33% bandwidth improvement as required by [21].

4.2 N x [H.264/AVC Single video SPTS] with Multiplexer

4.2.1 Prototype description

For the N x [H.264/AVC Single video SPTS] with Multiplexer approach, the prototype consists of N **H.264/AVC Single-video SPTS encoders** and an **MPEG Multiplexer**. The working principle of this approach is the following. The different H.264/AVC Single-video SPTS encoders generate AVC streams of the same content at different bit rates. The streams are synchronized in such a way that seamless switching between streams is possible. Then the output of all these encoders is passed to the MPEG Multiplexer, which is responsible for multiplexing the several Single-video SPTS into a unique Multi-video SPTS. In the remainder of the section each of the components is explained in more detail.

Figure 3 –OPTEC Encoding board

Figure 3 shows the prototype of the **H.264/AVC Single-video SPTS**. It is used for high quality real-time encoding of both SD and HD video via HD/SD-SDI, analogue (composite), and HDMI input interfaces. It has two 100Mbit Ethernet/IP ports, for transmitting the resulting output MPEG-2 TS over UDP. It supports the Simple Network Management Protocol (SNMP) for managing and fulfils synchronization of the different streams at different bit rates for the Multi-bitrate approach. In order to allow the PDD to seamlessly switch between the streams, encoders generate “closed GOP” video sequences with synchronized PTS/DTS timing information across streams. The resulting SPTS streams have synchronized PCRs as well. As these streams have the same GOP length, the PDD may easily switch from one stream to another, allowing decoders on the client side to show uninterrupted video. Additionally, a special module performs tagging of the random access points (RAP) by "priority boosting", i.e. the priority flag of the first MPEG-2 TS packet of a RAP is set to indicate that effective switching or access to the stream can be performed. This is explained more in detail for the next approach in section 4.3.1. The main features of the Single-video SPTS encoder are summarized in Table 3.

Table 3 - Main features of the H.264/AVC Single-video SPTS encoder

HD Encoding	**SD Encoding**
Video: ISO/IEC14496-10 (H.264/AVC) 1920 x 1080 59.94i/60i/50i/30p 1280 x 720 59.94p/50p 4:2:0 chroma sampling Encoding rate: 2 to 15Mbps Aspect Ratio 16:9 Downscaling from HD to SD	**Video:** ISO/IEC14496-10 (H.264/AVC) Baseline and Main Profile Level 3.0 720x480x29.97i 720x576x25i 4:2:0 chroma sampling Encoding rate: 150 Kbps to 6 Mbps
Audio:	**Audio:**

2 stereo pairs (embedded in SDI, Unbalanced in Analog) Sampling freq: 48 kHz AAC-LC (32 to 384kbps) Mono, stereo	2 stereo pairs Sampling freq: 48 kHz 16 bit support AAC-LC (32 to 384kbps) MPEG 1 L2 (56 to 384kbps) Mono, stereo

The **MPEG Multiplexer** is used to convert the Single-video SPTS solution into Multi-video SPTS so as to have a compliant and standardized solution towards the PDD. The input to the multiplexer is several Single-video SPTS of the same channel at different bit rates and it outputs a unique Multi-video SPTS. During this multiplexing operation, the PSI tables are changed accordingly, described in the following.

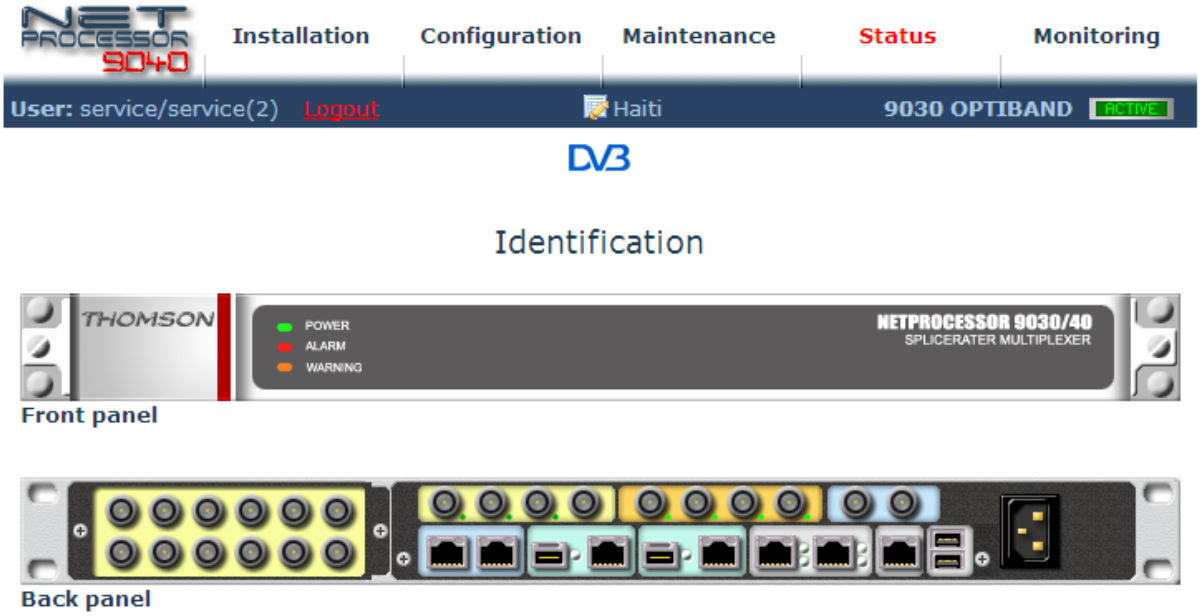


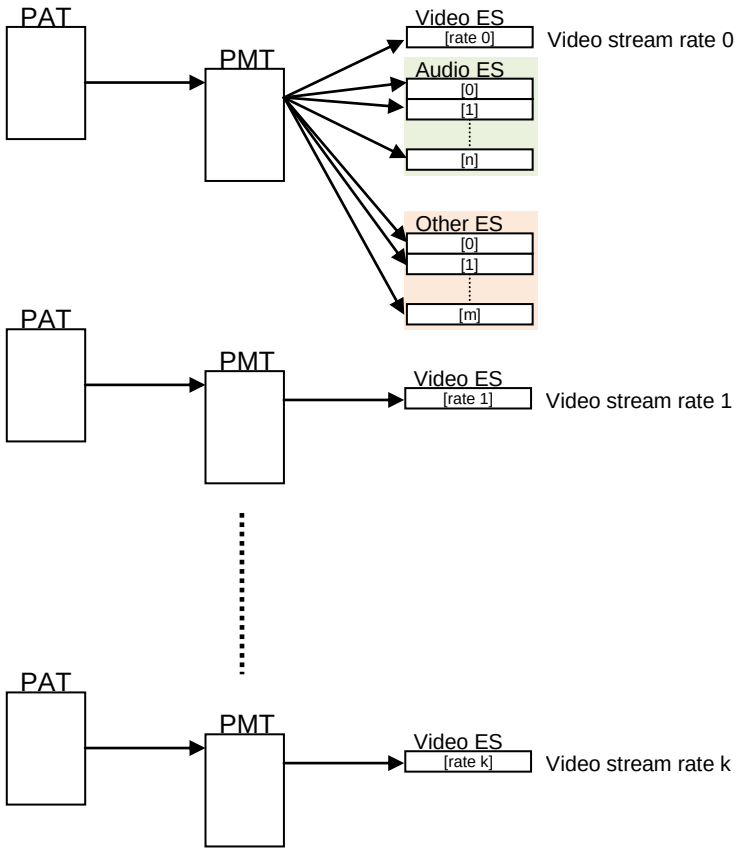
Figure 4 –TVN MPEG Multiplexer

Figure 4 shows the front panel and back panel of the multiplexer, where the input and output ports can be seen. The configuration for its IP inputs, IP outputs and multiplexing processes can be done as shown in the example in Figure 5. The Gigabit Ethernet setup is done for the different LAN RX according to the number of input Single-video SPTS provided. Set up is done for “Data in” IP LAN port and the required IP Multicast address. Gigabit Ethernet setup is done for the different LAN TX according to the number of output Multi-video SPTS produced. Set up is done for “Data out” IP LAN port and the required IP Multicast address.

Figure 5 –TVN MPEG Multiplexer Configuration example

In the following, the Single video SPTS and Multi video STPS are described, as well as the process followed by the multiplexer in order to convert several Single video SPTS into a single Multi video SPTS.

For Single-video SPTS solution, each video stream rate is transported over separate MPEG-2 TS container. The following figure illustrates the MPEG-2 TS tables to support this solution:



According to the figure above, each Single-video SPTS is transported separately and independently of Single-video SPTS that carries the other video bit rates of same channel.

The **MPEG Multiplexer** when converting the Single-video SPTS solution into Multi-video SPTS keeps only a unique Program Association Table (PAT) for the channel, the PAT has a single entry for single Program Map Tables (PMT). The PMT describes all the PIDs of the different video stream rates within a single table for the channel. If the multiplexer encounters video or audio or other ES PIDs conflict, a reallocation is done with new PIDs to solve the conflict.

The MPEG-2 TS distributes the Program Clock Reference (PCR) that is used for content synchronization at the decoder. For this purpose, the PCR, which is used for content synchronization at the decoder (e.g. video and audio synchronization), needs to be consistent across representations, which implies that the PCR has to be appended initially by the encoder at the adaptation field of each video stream of the different Single-video SPTS and has to be set identical for all streams of same channel (at different bit rates).

The different PCRs transmitted are corrected while multiplexing so as to deliver correct PCRs after the multiplexing operation for each video stream of the new built Multi-video SPTS.

The result of the configured multiplexing processing is a Multi-video SPTS. For Multi-video SPTS solution, all video streams of same channel are multiplexed into a single MPEG-2 TS container. The MPEG-2 TS is still considered as SPTS, because it transports a single program with different rate options. The following figure illustrates the MPEG-2 TS tables to support this solution:

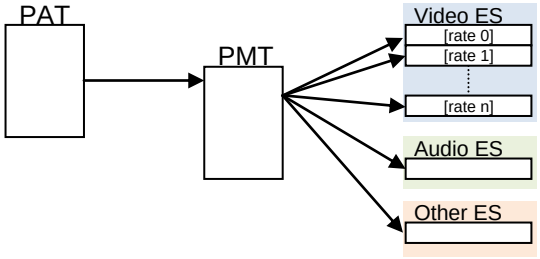


Figure 6 – Resulting Multi-video SPTS after MPEG Mux

The multiplexer complies with MPEG-2 and MPEG-4 standards. It supports MPEG-2 MP@ML, MPEG-2 MP@HL and MPEG-4-AVC Main and High Profiles for video and MPEG-1 layer 2/3, AC3 (Including 5.1), AAC-LC, HE-AAC audio standards. It is capable of multiplexing/remultiplexing up to 64 MPEG-2 TS received over ASI (asynchronous serial interface) and telecom interfaces, generating up to 6 multiplexes delivered over ASI/Gigabit Ethernet/ATM. For this purpose the multiplexer counts with two Gigabit Ethernet/IP ports that can be used for reception or transmission and two additional Gigabit Ethernet/IP ports only for output, being able to transmit a maximum of 520Mbps. The MPEG Multiplexer main capabilities are advanced management of PSI/SI/PSIP tables, service filtering and remapping, performing table injection/filtering from the Ethernet and TS inputs. A further capability of the multiplexer consists in duplication capability of the output stream. For management Web and SNMP interfaces can be used.

Table 4 – System interfaces

#	Source	Destination	Description	Format
1	Video input (live)	OPTEC Multi bitrate encoder	Video source with embedded audio	HD-SDI up to 1080i
2	OPTEC Multi bitrate encoder	TVN Multiplexer	Multiple channels with different bit rates H264 streams and associated audio encapsulated into different Single-video SPTS. All channels are synchronized to allow smooth change between the streams	IP/UDP SPTS
3	TVN Multiplexer	CSL PDD	Multiple channels with different bit rates H264 streams and associated audio encapsulated into Multi-video SPTS. All channels are synchronized to allow smooth change between the streams	IP/UDP SPTS
4	User (Admin)	TVN Multiplexer	Control and Monitoring (WEB GUI)	HTTP Multiplexer configuration
5	User(Admin)	OPTEC	Control and Monitoring application	Config file and Application

4.2.2 Head End prototype conformance check and system test

Figure 7 shows the test environment for the H264/AVC Single video SPTS prototype method that allows checking the correct operation of this prototype. This test environment consists of four main components:

a) The MPEG-2 TS over UDP/IP player
b) The H264/AVC Single video SPTS encoders
c) MPEG Multiplexer
d) The Multi-video SPTS over UDP/IP acquisition device

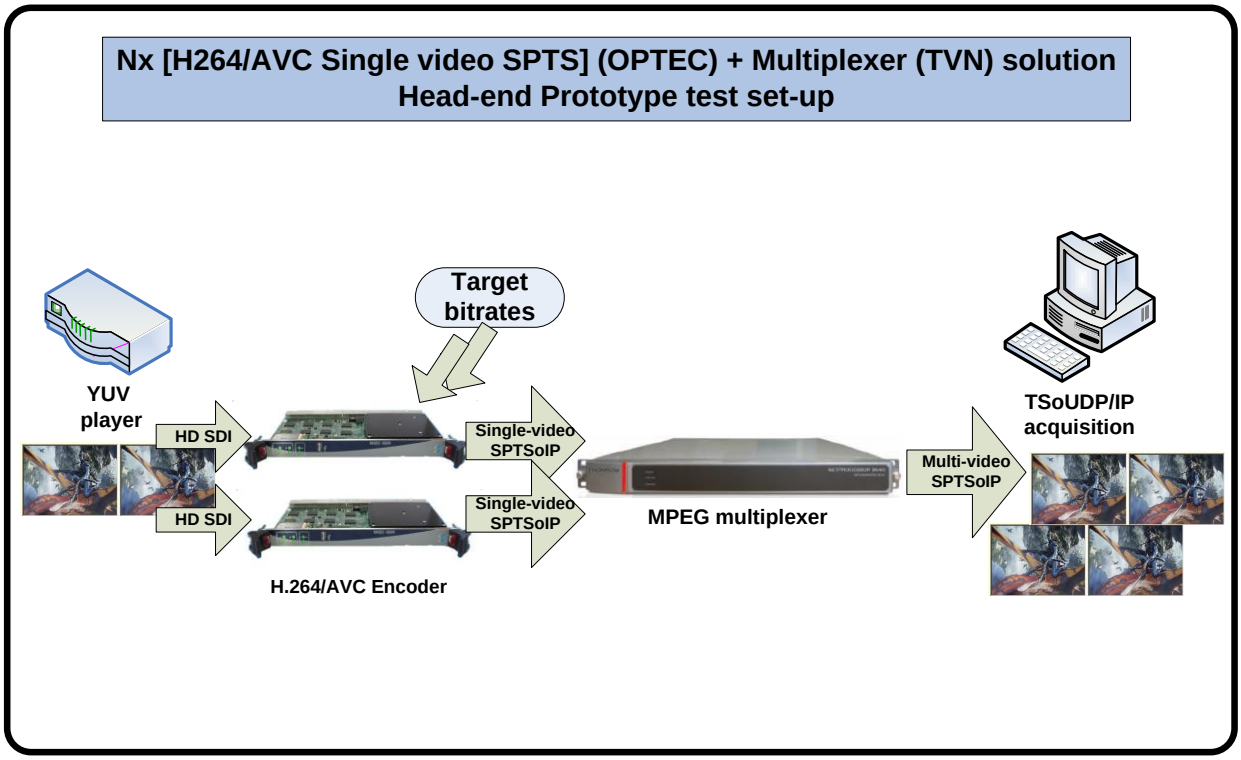


Figure 7 – Test environment for H264/AVC single video SPTS prototype

The output of the MPEG Multiplexer, i.e. the Multi-video SPTS, is sent to a PC capable of processing TS streamed over UDP/IP. It is checked that the created MPEG-2 TS is a correct Multi-video SPTS, by checking that the different versions contained in the stream can be decoded and presented, as well as the synchronization is well done, i.e. the different streams contain the RAP at the same frame position and the PCR are synchronized among the different video version.

Prototype results

With this prototype (i.e. N x [H.264/AVC Single video SPTS]+Multiplexer) the streams for the action clips and soccer clips at the aforementioned rates are created, as summarized in the following table.

Table 5 – Multi-video SPTS sequence for PDD interoperability

H264 Multi video SPTS									
Reference Clips	**Standard**	**Video mode**	**GOP length**	**Program number**	**PMT PID**	**video PIDs**	**Encoded rates Mb/s**	**Video Format**	**Multi video SPTS sequences files**
Action	H264	CBR	fixed	1	100	1001	8.0	HD 1920i 1080	OPT_MSPTSMUX.ts
						1002	6.5	HD 1920i 1080	
						1003	4.95	HD 1920i 1080	
						1004	3.45	HD 1920i 1080	

For D4.5.2 prototype, OPTEC provides two Multi-video SPTS sequence with four HD 1080i encoded streams (see bit rates in Table 2) with IDR-frames flagged for their first TS packet in order to validate the Packet Dropping algorithm (PDA).

Table 6 – Single-video SPTS sequence for STB interoperability

H264 Single video SPTS									
Reference Clips	**Standard**	**Video mode**	**GOP length**	**Program number**	**PMT PID**	**video PIDs**	**Encoded rates Mb/s**	**Video Format**	**Single video SPTS sequences files**
Action	H264	CBR	Variable	1	100	110	7,76	HD 1080i 1440	TVN_Docu_HD1440SinglevidSPTS_7.76.trp
Action	H264	CBR	Variable	1	100	110	7,76	HD 1080i 1440	TVN_Docu_HD1440SinglevidSPTS_7.76.trp

For D4.5.2 prototype, OPTEC provides two Single-video SPTS sequence with one HD 1080i encoded stream for STB interoperability.

4.3 H.264/AVC Multi-video SPTS with embedded Multiplexer

4.3.1 Prototype description

In this approach the prototype consists of a H.264/AVC Multi-video SPTS encoder with an embedded multiplexer that produces a Multi-video SPTS with multiple streams of the same content at different rates.

The input of this component is HD video up to 720p or 1080i via HD/SD-SDI and the output is a Multi-video SPTS over UDP, which means that all different versions of the same content are transmitted over the same port. For the Multi-video SPTS with embedded multiplexer a WEB GUI can be used for encoding configuration and management of the device.

The Multi-video SPTS encoder with embedded multiplexer is compliant with the following parameters in Table 7:

Table 7 – Video format for the Multi-video SPTS encoder

Standard	H264/AVC
Profile	High profile
Video format	1080i (1440) 50 @ 25 fps

Video mode	CBR
GOP structure	Adaptive
GOP length	Variable

The different characteristics of the different video versions in the Multi-video SPTS (i.e. the different streams) are defined in the Head End and include all video parameters like resolution, quantization, frame rate, pre-filters and so on. In order to achieve smooth playback of a stream at the decoder side, it is important to provide different video versions with different bit rates, to allow adaptation at the PDD when the available throughput in the network varies. However, these streams (video versions) within the Multi-video SPTS must "obey" the following encoding constraints Each of the streams within the Multi-video SPTS is encoded with adaptive GOP, with possibility of applying scene detection for determining the size of the adaptive GOPs, keeping GOP alignment among the different streams at different bit rates. Furthermore, all streams should be compliant with the constraints on the MPEG Transport Stream System Target Decoder (T-STD) buffers. The systems layer of the MPEG-2 standard specifies transport stream system target decoders (T-STD). The purpose of a T-STD is to ensure that transport streams that have passed this “theoretical” decoder can be decoded with any true decoder. T-STD buffer model embodies the timely and controlled delivery of data. Therefore, synchronization across the TS of T-STD levels before each IDR-frame is required not to violate the buffer constraints. For the Multi-video STPS only one audio is sent within the multiplex.

In the following, the main aspects of the Multi-video SPTS approach are explained in terms of how to create switchable content and indicate the switching points to the PDD. For both multi-rate approaches based on H.264/AVC, switching between the different bit rate streams is only done at instantaneous decoding refresh (IDR) frame boundaries. IDRs are well defined RAP that allow for switching from one video stream at a given bit rate to another. IDRs are the beginning of a coded video sequence. In order to allow the multi bitrate approach for effective seamless switching, it requires that all H.264/AVC streams of same source are IDR-frame synchronized. Further, the same frame has to be encoded by all encoders and the IDR-frame shall start at the same time to allow the Packet Dropping Algorithm (PDA) to switch between the different bit rate streams transparently to the client.

The H.264/AVC encoded streams are encapsulated with MPEG-2 TS in an embedded multiplexer, resulting in a Multi-video SPTS, which description is provided in [2] in section 3.3.1.1. Figure 6 shows the working principle of an MPEG-2 TS multiplexer, where different video stream, audio streams and metadata is passed to the multiplexer and the latter generates a single MPEG-2 TS.

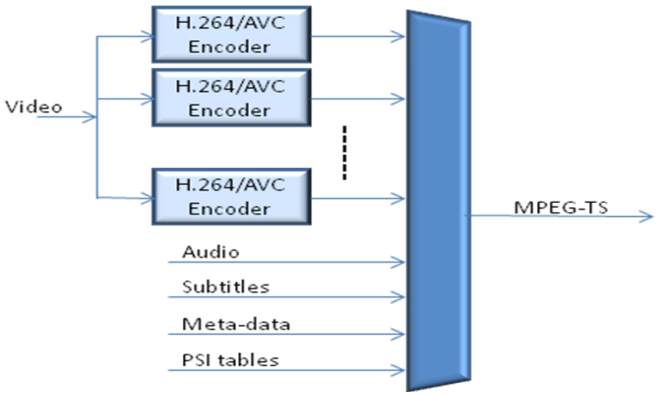


Figure 8 - MPEG-2 TS multiplexer

In Figure 9 the whole process of compression and encapsulation is shown. First the video is compressed into an elementary stream. Then, it is encapsulated into the packet elementary stream (PES), where each frame is typically assigned to a PES packet and all NAL units of this frame are encapsulated in a single PES packet with a header containing some important information such as presentation time or decoding time. Finally, these PES packets are split into transport stream (TS) packets of 188 bytes. The TS header contains some important information for the client, as e.g. to allow de-multiplexing and filtering of programs. Note further that each MPEG-2 TS packet only contains data from a single PES, there is no TS packet with data from two different PES packets.

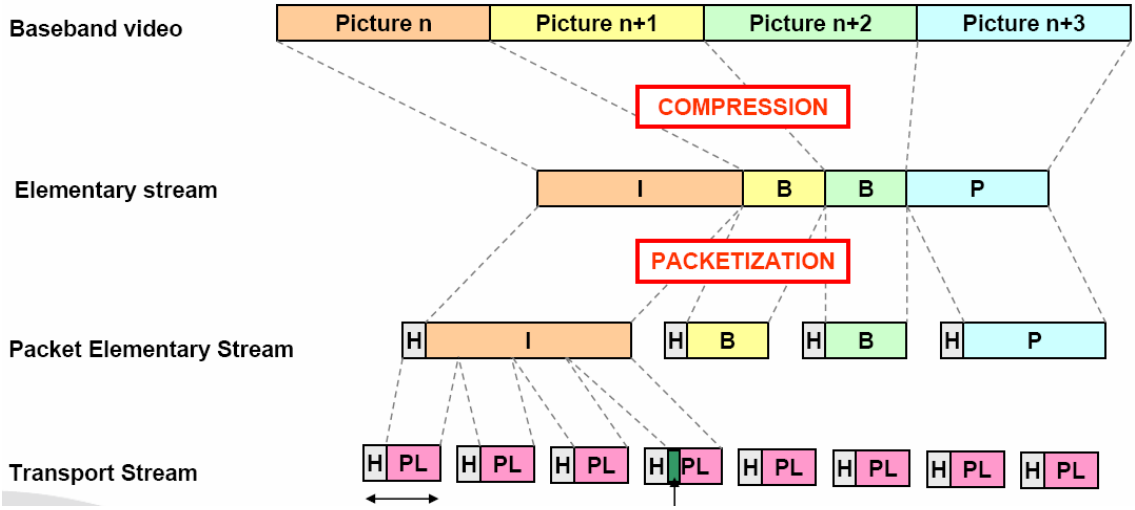


Figure 9 – MPEG-2 transport video packetization

In order to clearly identify Random Access Points (RAP) for the PDD, the transport priority flag in TS packet header is used (see Figure 11). RAPs are points in the stream, where seamless switching can be done. A RAP consists of a Sequence Parameter Set (SPS) NAL unit, a Picture Parameter Set (PPS) NAL unit and an IDR-frame. Since an IDR breaks the dependency of following frames to frames previous to the IDR, it is a perfect solution for applying seamless switching from one stream to another. However, new SPS and PPS are needed since a Sequence Parameter Set (SPS) contains parameters applicable to a video sequence and a Picture Parameter Set (PPS) contains parameters applicable to a picture that are necessary for decoding. Therefore the encoders have to flag the first TS packet of the RAP, which contains the SPS since this precedes both the PPS and the IDR, indicating that this first packet can be used as a switching point. For simplicity, in Figure 10, the priority flagging points to the IDR, while it actually points to the first TS packet that contains the SPS NAL unit packet prior to the IDR frame, since video slices refer to PPS and SPS transmitted before them. As already mentioned, at the beginning of a coded video sequence there is an IDR access unit. An IDR access unit contains an intra coded picture (that can be decoded without decoding any previous pictures in the NAL unit stream), and the presence of an IDR access unit indicates that no subsequent picture in the stream will require reference to pictures prior to the intra picture it contains in order to be decoded.

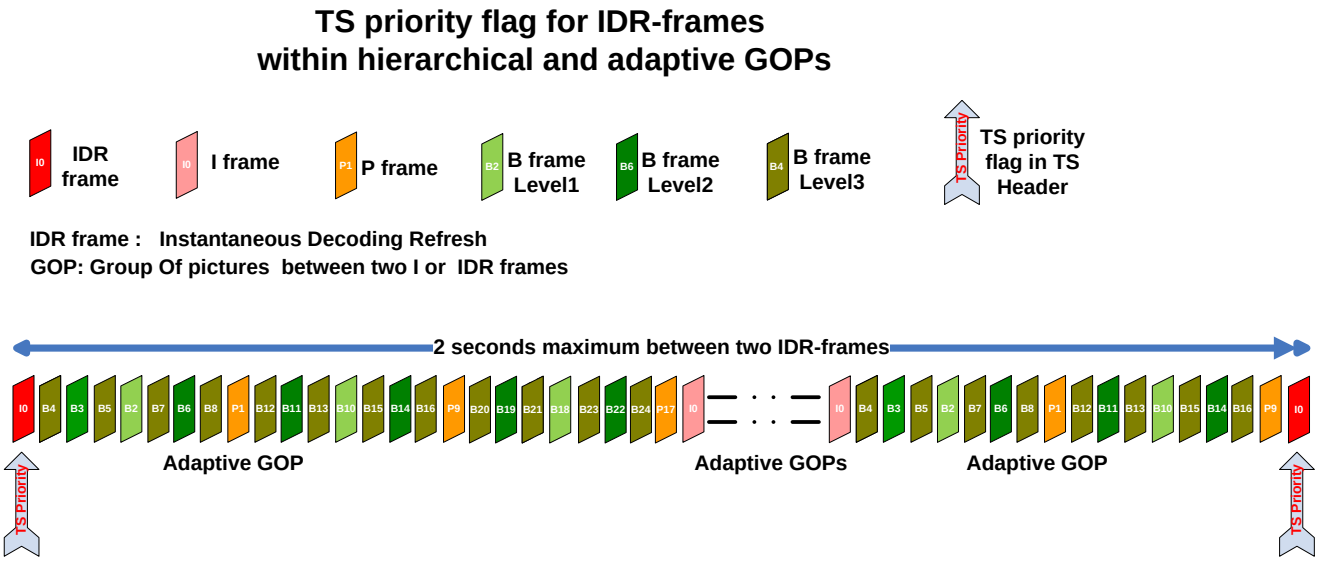


Figure 10 – Adaptive GOPs with TS priority flag for IDR-frames

The H.264/AVC Multi-video SPTS encoder allows tunable interval between IDR-frames and Transport priority flag to comply with OPTIBAND GOP size: between 1.5 to 2 seconds approximately.

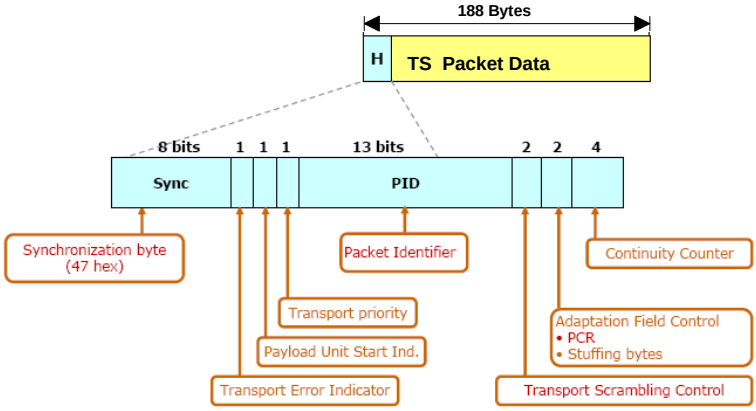


Figure 11 – Transport priority flag in TS Header

Further important information for the multi bit rate approach that deserves being mentioned is the following MPEG-2 TS PSI tables: Program Association Table (PAT), which has the PID value of 0x0000 and lists the PIDs of the Program Map Tables (PMT) for all programs, the PMT itself, which defines the set of PÌDs associated with a program, and the Conditional Access Table, which provides information for conditional access. Figure 10 shows a typical example of these tables for a multiplex with 4 programs (PAT with 4 entries) having one of the programs three components: video audio and data.

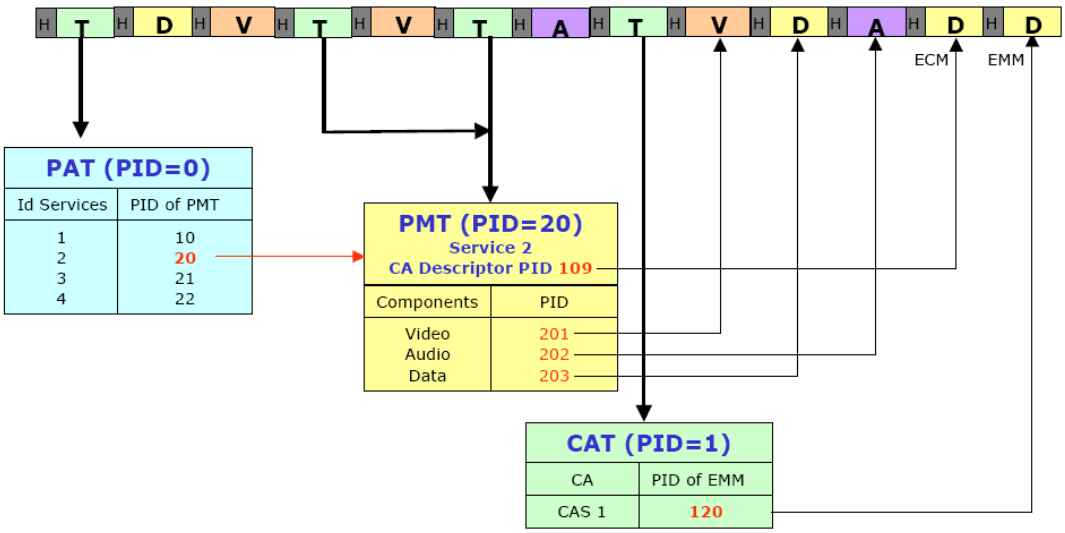


Figure 12 – PSI Tables example

For the Multi-video SPTS approach, all different video versions of the same content (same channel) at different rates are multiplexed into a single program. Typically, the different versions would be encapsulated into different programs resulting in a Multi Program Transport Stream (MPTS) but for this approach all versions are transported in a single program with different rate options, which will be adapted at the PDD, resulting the switching from one stream to another transparent for the client (STB). Therefore, it is called Multi-video SPTS solution. Figure 11 illustrates the MPEG-2 TS tables to support this method:

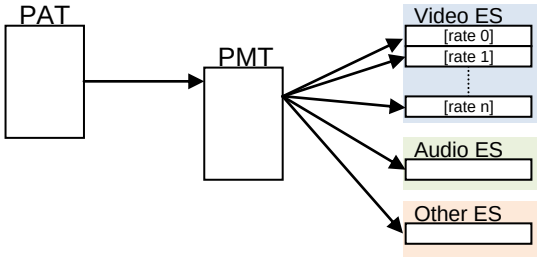


Figure 13 – Multi-video SPTS solution.

As seen in the figure, the PAT has a single entry since there is only one program, and the PMT has several entries for video, one per video stream rate. This is even clearer in Figure 12. By transporting all video streams of same program on a single TS/UDP, we can keep a single audio elementary stream (ES) and single subtitle ES (or one per supported language) for all video streams, no need for unnecessary duplication as done for the video ES. For Multi-video SPTS, the PAT table specifies that the MPEG-2 TS carry a single program, meaning a single PMT table. The PMT table specifies all elementary streams within the transported program, as illustrated by the following tables:

PAT (PID = 0)

Program	PID
0	PMT_PID

PMT (PID = PMT_PID)

StreamType	PID
Video rate 0	VID_PID_1
Video rate 1	VID_PID_2
…	…
Video rate k	VID_PID_K
Audio	AUD_PID
PCR	VID_PID_1

Figure 14 – MPEG-2 transport stream Program Specific Information

In order to allow the PDD seamlessly switching from one representation to another, the streams in the Multi-video SPTS approach must fulfil some multiplexing constraints. The PCR is not carried in its own private PID but it is carried on the video PDD, referenced in the PMT as shown in the example in Figure 12. Furthermore, the arrival time of PCR at the PDD shall have a jitter less than ±200ms and PCR and PTS values are synchronized across streams, i.e. the run at the same rate for every "sub-channel" (stream rate), being the PTS of equivalent frame for each version (frames of the same picture) the same.

As seen in Figure 12, each of the streams at different rate has a different and unique PID that is specified in the PMT table of the channel. The PDD may decide to switch from one stream to another at RAP (IDR boundaries). As explained in [3], once the PDD decides which version of the data to send to each STB, the PID field of the MPEG-2 TS of the video may be changed (if necessary) to the pre-configured value in order to keep a constant video PID for the STB. This pre-configured value is selected to be one of the multi bit rate streams PID so that the PMT does not have to be changed. Therefore, the STB does not notice that there has been a switch between representations.

For this purpose, the PCR, which is used for content synchronization at the decoder (e.g. video and audio synchronization), needs to be consistent across representations, which implies that the PCR has to be appended at the adaptation field of each video stream and has to be set identical for all streams of same contents (at different bit rates). Consequently, PTS and DTS timestamps are also set to the same value at same pictures in all streams. Another important field of the TS header that has to be considered when switching among representations is the continuity counter (CC), which is a 4-bit field incrementing with each Transport Stream packet with the same PID that wraps around to 0 after its maximum value. The CC shall not be incremented when the adaptation_field_control (AFC) field of the packet equals '00' or '10'. However, if this is not the case it should increment and be consistent as defined in ISO/IEC-13818-1. The Head End does not handle CC synchronization. It is the PDD the one responsible for modifying the CC field of the MPEG-2 TS header of the forwarded stream per destination (STB) in order to be compliant with ISO/IEC-13818-1.

Table 8 – System interfaces

#	Source	Destination	Description	Format
1	Video input (live)	TVN Multi bitrate encoder with baseband interface	Video source	HD-SDI up to 720p or 1080i
2	Audio input (live)	TVN Multi bitrate encoder	Audio source	IP/UDP TS No constraints
3	TVN Multi bitrate encoder	CSL PDD	Multiple channels with different bit rates H264 streams and associated audio encapsulated into Multi-video SPTS. All channels are synchronized to allow smooth change between the streams	IP/UDP SPTS
4	User (Admin)	TVN Multi bitrate encoder	Control and Monitoring (WEB GUI)	HTTP Encoder configuration

4.3.2 Head End prototype conformance check and system test

Figure 13 illustrates the test environment for the H264/AVC Multi-video SPTS prototype method. This test environment consists of three main components:

a) The TS over UDP/IP player
b) The H264/AVC Multi bitrate encoder with embedded Multiplexer
c) The Multi-video SPTS over UDP/IP acquisition device

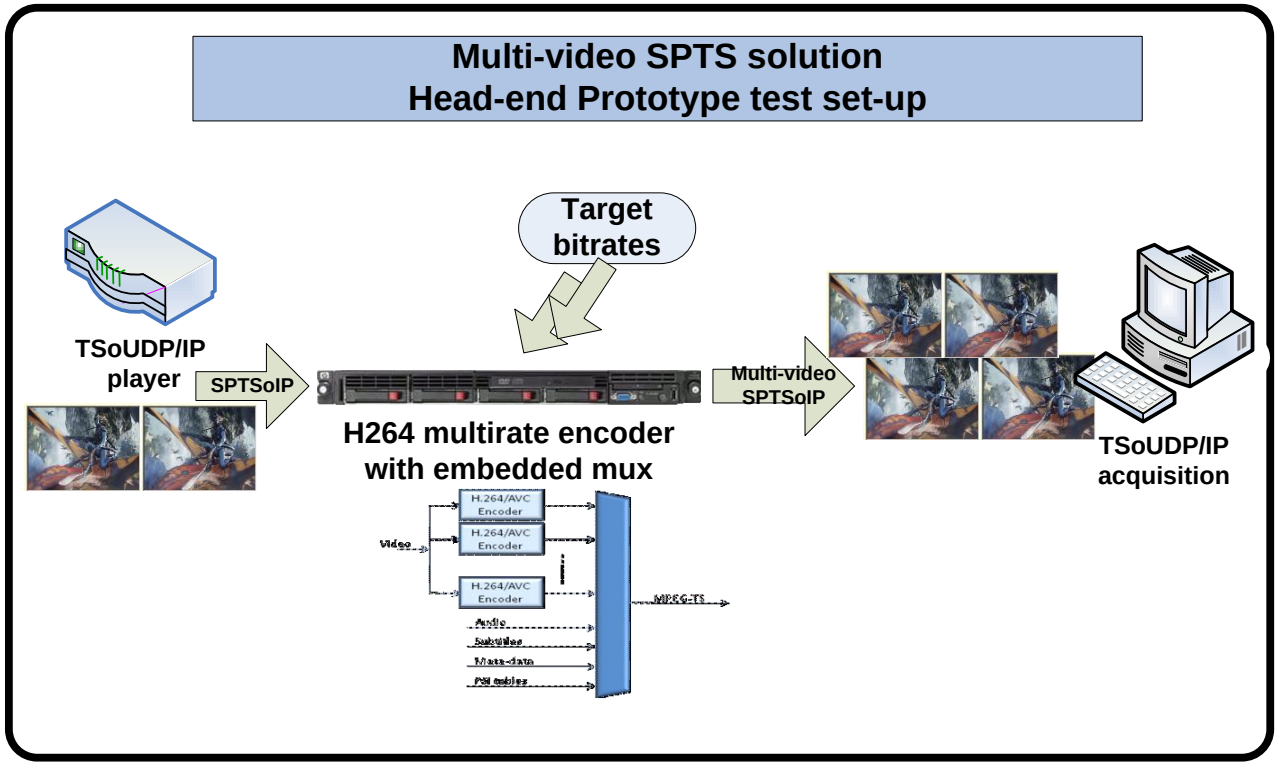


Figure 15 - Test environment for H264/AVC Multi-video SPTS prototype

The output of the Multi-video encoders (with embedded mux) i.e. the Multi-video SPTS, is sent to a PC capable of processing TS over UDP IP. It is checked that the created MPEG-2 TS is a correct Multi-video SPTS, by checking that the different version (with different video rate) contained in the stream can be decoded and presented, as well as the synchronization is well done, i.e. the different streams contain the RAP at the same frame position and the PCR are synchronized among the different video version.

Prototype results

One sequence is encoded according to the defined bit rates parameters at the application layer so as to provide SPTS files to be streamed towards the PDD. In this case, the stream of type documentary is encoded.

The tables below list the properties of video sequences which are used for testing of the H264/AVC Multi-video SPTS-based Head End.

Table 9 – Multi-video SPTS sequence for PDD interoperability

H264 Multi video SPTS									
Reference Clips	**Standard**	**Video mode**	**GOP length**	**Program number**	**PMT PID**	**video PIDs**	**Encoded rates Mb/s**	**Video Format**	**Multi video SPTS sequences files**
documentary	H264	CBR	variable	1	100	110	7,76	HD 1080i 1440	TVN_Docu_HD1440MultividSPTS_7.76_4.8.trp
						210	4,8 (-38%)	HD 1080i 1440	

For D4.5.2 prototype, TVN provides one Multi-video SPTS sequence with two HD 1080i (1440) encoded streams (bw_level_0 and bw_level 2 in Table 2) with IDR-frames flagged for their first TS packet in order to validate the Packet Dropping algorithm (PDA). Only two streams are provided due to limitations in the capability of encoders, which are able to produce two streams each. However, some flexibility for the Live Multi-video Encoder solution exists and studies on going when this deliverable is released show that the 3x2 streams Multi-video SPTS configuration (3 encoders and 2 streams per encoder) could be replaced without changing the hardware by a 1x4 streams Multi-video SPTS configuration, for instance. In this case, then, four streams of the same content at different bit rates could be generated, creating thus the four bandwidth levels reported in Table 2.

Table 10 – Single-video SPTS sequence for STB interoperability

H264 Single video SPTS									
Reference Clips	**Standard**	**Video mode**	**GOP length**	**Program number**	**PMT PID**	**video PIDs**	**Encoded rates Mb/s**	**Video Format**	**Single video SPTS sequences files**
documentary	H264	CBR	variable	1	100	110	7,76	HD 1080i 1440	TVN_Docu_HD1440SinglevidSPTS_7.76.trp

For D4.5.2 prototype, TVN provides one Single-video SPTS sequence with one HD 1080i(1440) encoded stream for STB interoperability.

The tools used for conformance check are:

- Wireshark for our IP streams live capture
- Internal tool to extract the TS file from the IP encapsulation
- VLC to play the TS video stream to control the correct streaming from our encoders

- Internal analyzer to identify Random Access Points (RAP) in the transport priority flag in TS packet header, PCR synchronization analysis, T-STD simulation, and TS metadata analysis (such as Program number PID , PMT PID , Video PIDs, Audio PIDs).

4.4 Encryption and decryption component

Testing the solution that is based on AVC encoding with multiple profiles requires use of a standalone scrambler application. A scrambler application takes the input MPEG-2 TS, scrambles the contained programs and includes ECMs in the stream for the descrambler. The output of the scrambler is a scrambled MPEG-2 TS. The descrambler application will take the scrambled TS, extract the ECM information and descramble the contained programs. The output of the descrambler is a clear TS, which means that the output of the descrambler (at the STB) is equal to the input TS to the encryption component at Head End with the exception of the added ECM packets and possible altered PSI table information (the scrambler adds the CA Descriptor, thus modifying the tables).

The scrambler and descrambler applications may use a file or a UDP connection as an input and it outputs data to a file or UDP connection. The scrambler and descrambler applications are command line tools with the following command arguments

- Input (-i)
  - *filename* or *udp://ip:port*
- Output (-o)
  - *filename* or *udp://ip:port*
- Bit rate (-b)
  - *bits-per-second*

The scrambler application requires the bit rate of the incoming stream to calculate the ECM injection rate and Control Word period. Taking an appropriate value from the PCR contained in the stream will add a significant delay in the moment at which scrambling can start.

4.5 Set-Top-Box

Since video decoding is a demanding activity in terms of computing power, all modern STBs have dedicated hardware to do it. This enables hardware manufactures to incorporate less powerful general purpose processors to control the rest of the system, to reduce the costs of the hardware in comparison with general purpose computers.

Most STB architectures will be based in MIPS or ARM processors, with specific processors for video and, optionally, audio decoding.

Thus, it is not possible to adapt anything in the decoding process without leading to high costs, since that will force any operator to replace all their STBs (at least one per end user). For this reason, the multi-rate solution must ensure that the streams the STB will receive comply with the constraints imposed by AVC standard profiles.

In Figure 14 the blocks of a hardware video decoder are shown. The main components are listed and explained below:

- A transport stream demuxer. This unit reads transport stream packets from a FIFO queue that could be fed either by the CPU (e.g. from a file) or by other input devices using DMA (e.g. a DTV tuner). The demuxer also sets the system clock reference and corrects its deviations from the timestamps received in the transport stream.
- The System Clock Reference (SCR). A 90KHz clock used to synchronise the video and audio outputs and, in case of live streaming, to synchronise with the streamer clock.
- A video decoder. The video decoder collects coded frames from a FIFO buffer connected to the demuxer, decodes them and stores the decoded pictures in the picture buffer.
- An audio decoder. This unit is optional since decoding audio might be done by software in the CPU.
- Video Output. This is roughly equivalent to a regular computer video card, but customised for the special case of STBs. It controls different kind of outputs to connect to TV sets, rather than computer monitors, such as HDMI, components, S-video or composite. Also, it usually organises the frame buffer in layers (see Figure 15), so that the video engine can write output frames in one layer and the

CPU can write generated graphics in other layer independently. The video output controller will blend them to generate the actual output signal.

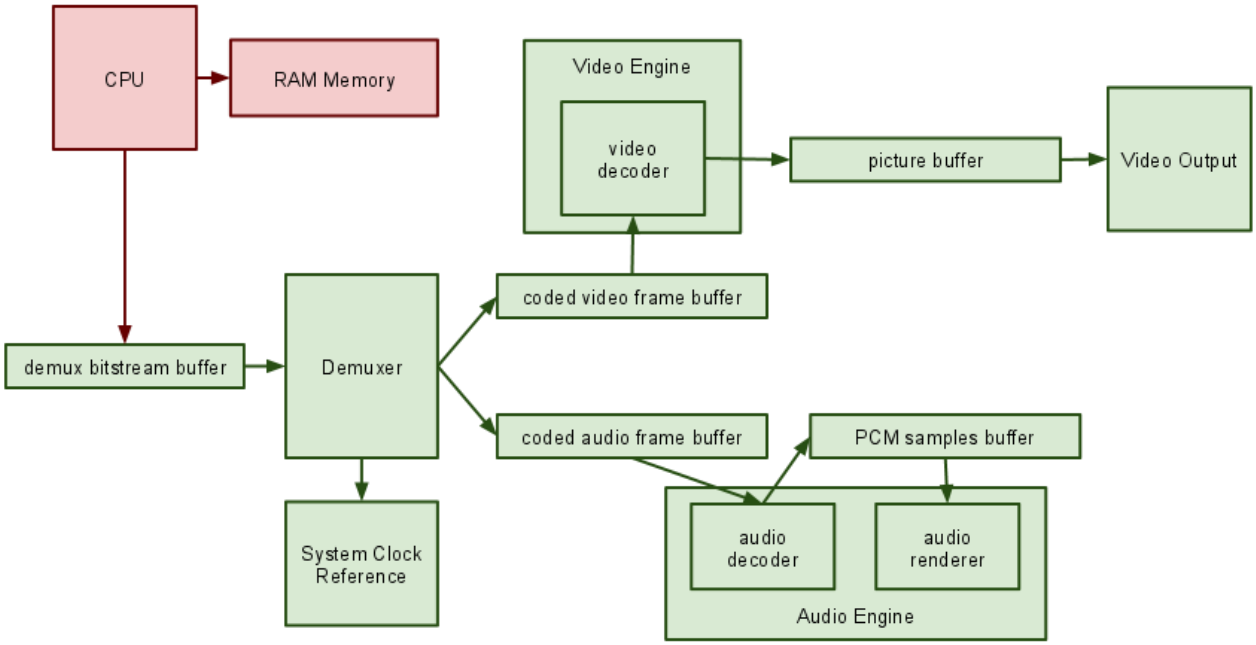


Figure 16 – Hardware Video Decoding Blocks

This architecture has the advantage that the software in the CPU can run quite independently from the video decoding and output rendering process. On the other hand, the main disadvantage is that the demuxing, decoding, and playback process is rigid and prone to synchronisation errors. If the input stream bit rate is bursty, buffers in the decoding chain may overflow or underflow, leading to video jerkiness or even to a crash of the video unit that would require a complete system reboot.

In an IPTV setting, the STB video player follows next steps once a user tries to tune in a multicast channel:

1. Open an UDP socket and join the multicast group
2. Initialise the decoding hardware. If the decoder supports multiple codecs, this step involves obtaining that information from somewhere. It could be detecting that in the input video stream, getting it from the middleware metadata, etc.
3. Read data from the socket in a buffer
4. Transfer the buffer to the demuxer and loop to 3.

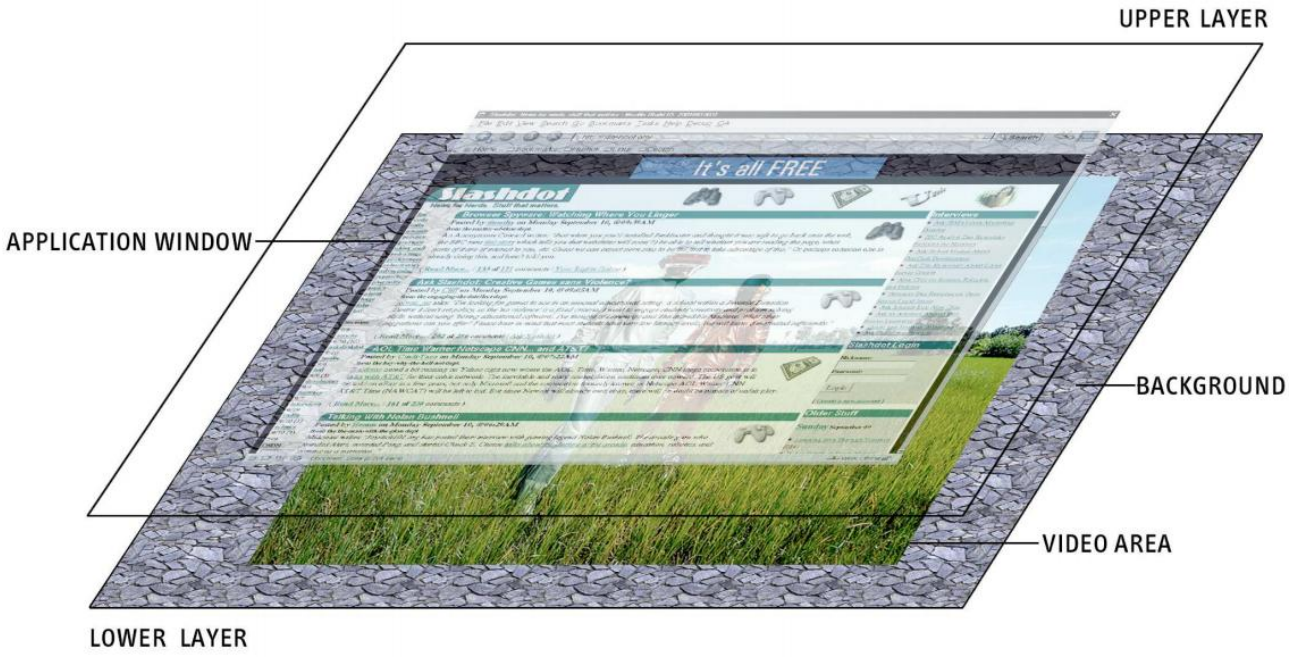


Figure 17 – *Multilayered Graphics Output*

Most STB boards will have special memory buses to transfer data to the video processor to minimise the transmission burden from the network card controller. There are, thus, several buffers involved in video playback. The CPU uses the reader buffer to absorb possible jitter from the network. Once it starts sending the stream to the video processor, the clock synchronisation mechanism starts and the video processor begins to move data across its internal buffers, assuming the buffering constraints imposed by the MPEG standards will hold.

There are several issues that can arise as e.g. listed below:

- If the network stalls for too long, the CPU buffer will empty and will fail to deliver data to the video engine at the required rate. The video engine will also stall, and the user will observe video artefacts. In this case, the most likely outcome is jerkiness due to images freezing in the screen
- If UDP packets are lost in the network, the CPU won't realise that, passing damaged input to the video processor. Different video engines will deal differently with this problem, but, inevitably, the user will perceive much lower quality due to lost frames or decoding artefacts.
- If the video server clock is not synchronised with the STB clock, and the video engine fails to synchronise it, the input rate will be slightly, but consistently, higher or lower than optimal, leading to internal video processor overflows or underflows. Underflows will lead to jerkiness, since the playback has to stop to refill the buffers and start playing again. Overflow effect may be damaged output or jerkiness, depending on how the video engine reacts to that problem.

To allow end users to access the video decoding functionality, vendors provide STBs with an application layer that runs in the CPU. That application layer will control the playback process described before, but also draw the graphical user interface (using the multilayer capabilities of the graphics output). This could be a standalone application for consumer electronics STBs that users buy for general purpose usage, or IPTV middleware client that connects to the operator middleware server, as explained in previous sections.

In order to be compatible with standard STBs, the multi-rate approach would have to ensure the timing constraints imposed by the MPEG standards. That is possible in synchronous networks such as DTV, Cable or Satellite. In IP networks it's impossible to ensure a fixed delay, so in practice this synchronisation constrains are relaxed in favour of bigger buffers. It is also required that quality of service in the network ensures very low packet loss rates (around one packet loss per minute at most).

For the prototype, an off-the-shelf Blusens Web:TV set-top-box will be provided. The Web:TV box is based on a Sigma Designs SMP8650 series Sistem-on-Chip and follows the above general characterization. Its main processor is a MIPS compatible 24Kf core, supported by specialized audio, video and security processors. The general architecture is very similar to existing set-top boxes already commercially deployed by Telecom Italia, using the SMP8634 processor which is a direct ancestor of the SMP8650.

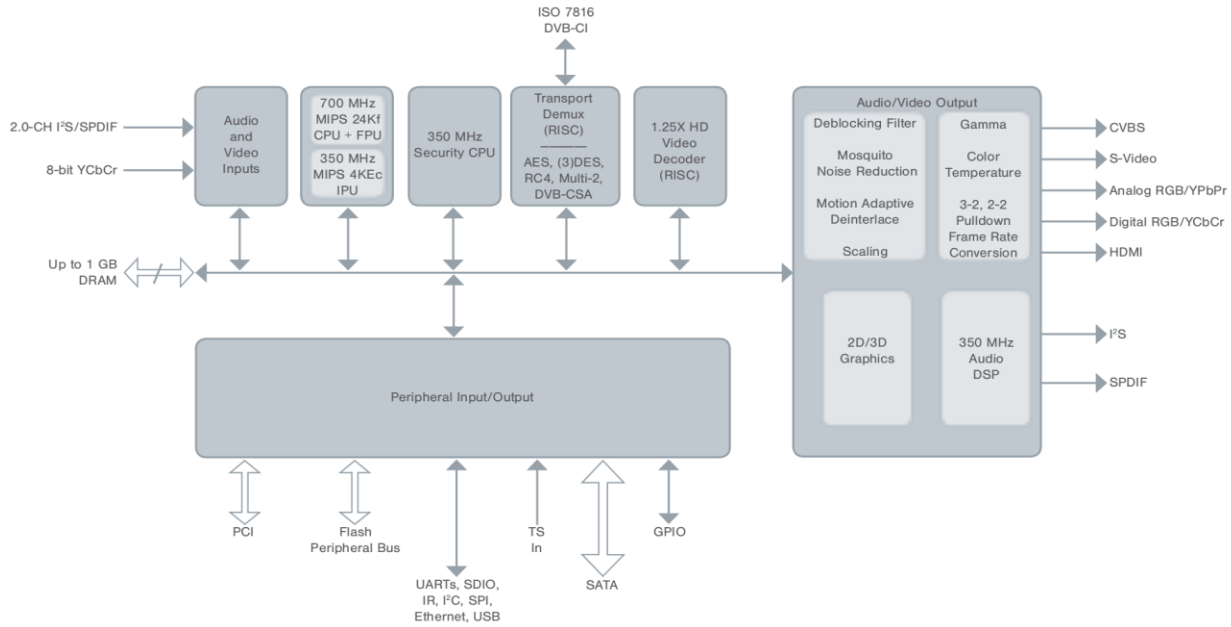


***Figure 18* – Main components in the Web: TC set-top-box system-on-chip**

From a hardware viewpoint, data paths for the incoming MPEG transport streams are almost the same whether the source is a DVB-T tuner and demodulator module (usually connected over an SPI bus) or a UDP stream received via Ethernet. However, on the software side, they are quite different. For an application running on the main processor core, playing a Transport Stream received from DVB-T means creating a (purely software) connection from the SPI connected DVB-T module to the transport demux, and then pumping packets – a task that implies minimal buffering, as the Multiple Program Transport Stream is received as a strictly constant bit rate flow over a synchronous transport (the DVB-T mux).

On the other hand, receiving a stream over UDP involves the operating system IP stack. Ethernet, IP and UDP are all asynchronous protocols with no timing guarantees of any kind; in practice, the Ethernet driver will perform some buffering (minimal: mostly for the purpose of interrupt mitigation), and the IP stack will buffer packets while moving and checking them. All of this adds to the packet jitter imposed by the possibly long asynchronous path between the IPTV Head End and the STB Ethernet interface.

As a consequence of the above, there is a significant latency before an UDP stream starts playing, and the effect of some missed packets in the Transport Stream are not exactly the same whether it comes from Ethernet or from a DVB-T module. For the benefit of the end-user experience, all Head End components should be careful not to add any significant latency due to the introduction of the multi-bitrate approach.

5. SVC-based Head End prototype

5.1 Introduction

In this approach, the Head End prototype consists of five main software components, namely: the *SVC Encoder*, the *MP4 File Creator*, the *MPEG-2 TS Encapsulator*, the scrambler and *RTP Encapsulator*. The SVC test bench is shown in Figure 19 and describes the technical basis of all the technologies involved in the SVC-based Head End processing. Figure 19 shows the main components of the SVC-based IPTV solution for packet dropping like the Head End, the Packet Dropping Device and the STB emulator for SVC connected to an HD display. These components are used in order to achieve the objective of the project to reduce the bandwidth of HD video by 33%.

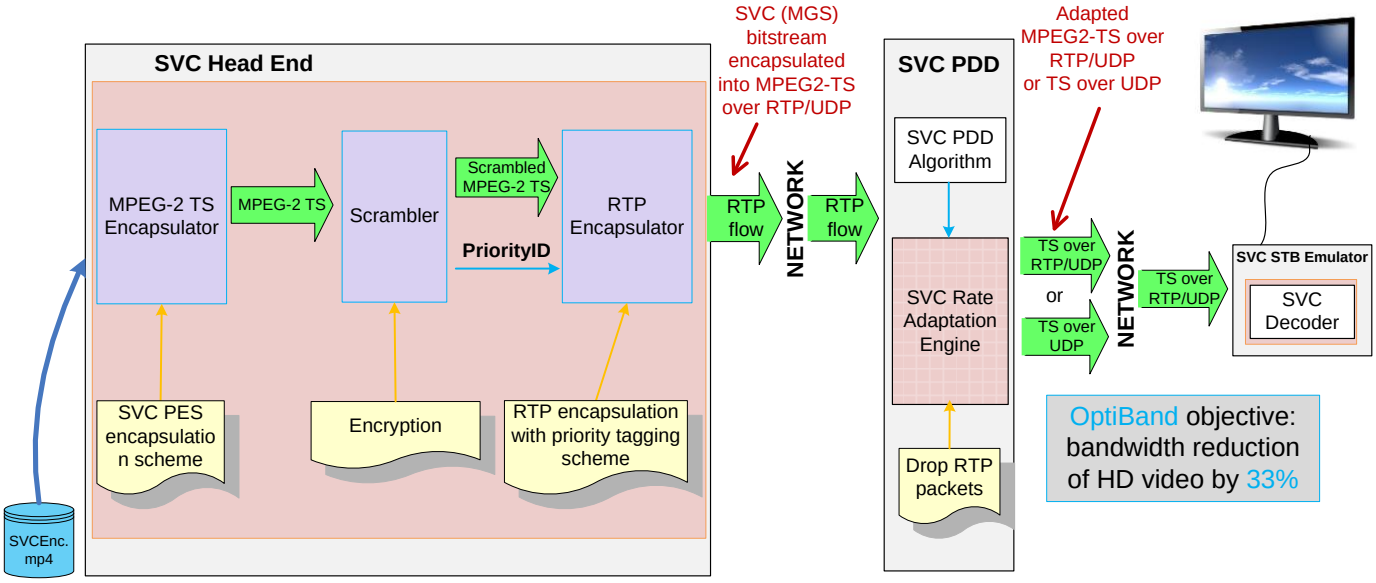


Figure 19 – SVC-based IPTV solution for packet dropping

In order to implement the SVC-based solution for optimisation of bandwidth for IPTV video streaming the Head End produces SVC content encapsulated in MPEG-2 TS and further in RTP with priority tagging for providing some signalling mechanism to the PDD in order to allow the PDD to forward a video quality (version) that best matches the capabilities at the network, i.e. a video with a rate that matches the available throughput. The Head End provides to the SVC-based Packet Dropping Device several valid H.264/SVC bit streams (one per multicast channel). The SVC-based Packet Dropping Device can either keep or drop a complete RTP packet to adapt HD video streams. The metadata needed for the decision whether to keep or drop an RTP packet is signalled using the SSRC field in the RTP header. As can be seen in Figure 19 two possible outputs of the SVC PDD are considered: TS over RTP/UDP and TS directly over UDP. These output streams are transmitted to certain STB (PC-based STB emulator in case of SVC) to decode the SVC bit stream, as shown in Figure 19.

The SVC-based Head End is divided in two main functional modules as presented in Figure 20: **SVC Encoder and MP4 File Creator** and **MPEG-2 TS and RTP Encapsulator.**

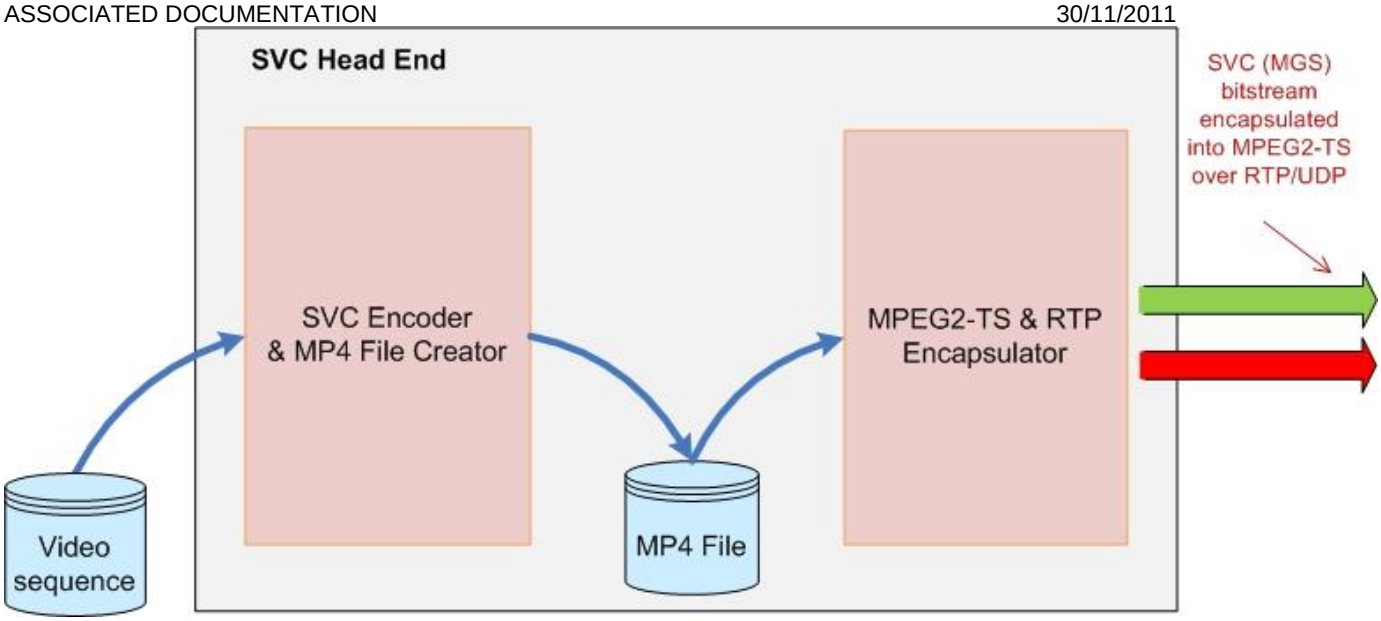


Figure 20 – SVC Head End

5.2 Main Head End components

5.2.1 SVC Encoder and MP4 File Creator

Figure 21 illustrates the functionality of the encoding process and mp4 file creation for the video clips "CC1_Action_ClipA", "CC1_Action_ClipB" and "CC1_Action_ClipC". The Joint Scalable Video Model (JSVM) software [15] (the reference software for the Scalable Video Coding standard) is used for encoding a video sequence given in the YUV format. Necessary encoding parameters are specified by several configuration files. After encoding, the SVC bit stream is encapsulated into MP4 file format. For this task, an MP4 File Creator for SVC is used. The MP4 File Creator is using an XML configuration file. The resulting MP4 file will be used for further processing.

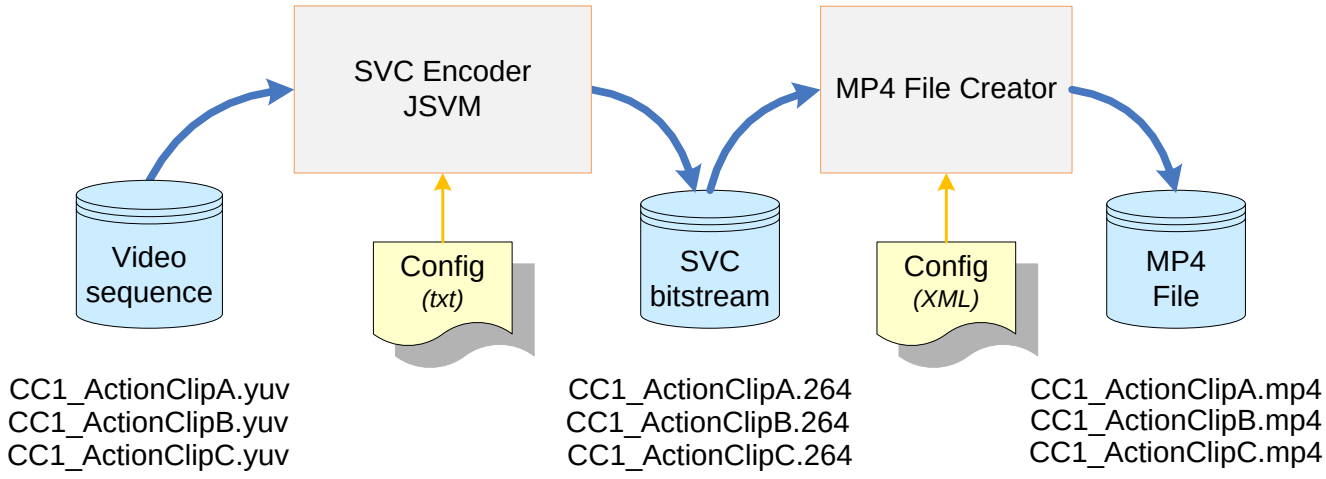


Figure 21 – SVC Encoder (JSVM) & MP4 File Creator

Table 11 lists properties of video sequences which are used for testing of the SVC-based Head End.

Table 11 – Video format of streams for the SVC-based Head End simulation

Video sequence	Standard	Content	Max bit rate	Video resolution	Frame rate (fps)	Video mode	GOP length
CC1_Action_ClipA	H.264/SVC	Action movie	7 ÷ 8.5 Mbps	1920x1080	25	VBR	8 (Intra frame every 24 frames)
CC1_Action_ClipB	H.264/SVC	Action movie	7 ÷ 8.5 Mbps	1920x1080	25	VBR	8 (Intra frame every 24 frames)
CC1_Action_ClipC	H.264/SVC	Action movie	7 ÷ 8.5 Mbps	1920x1080	25	VBR	8 (Intra frame every 24 frames)

5.2.2 MPEG-2 TS and RTP Encapsulator

Figure 22 illustrates the functionality of the encapsulation and the additional tagging process of SVC data using RTP header. MPEG-2 TS and RTP Encapsulator have been developed by Fraunhofer HHI as plugins for the VLC media player [8].

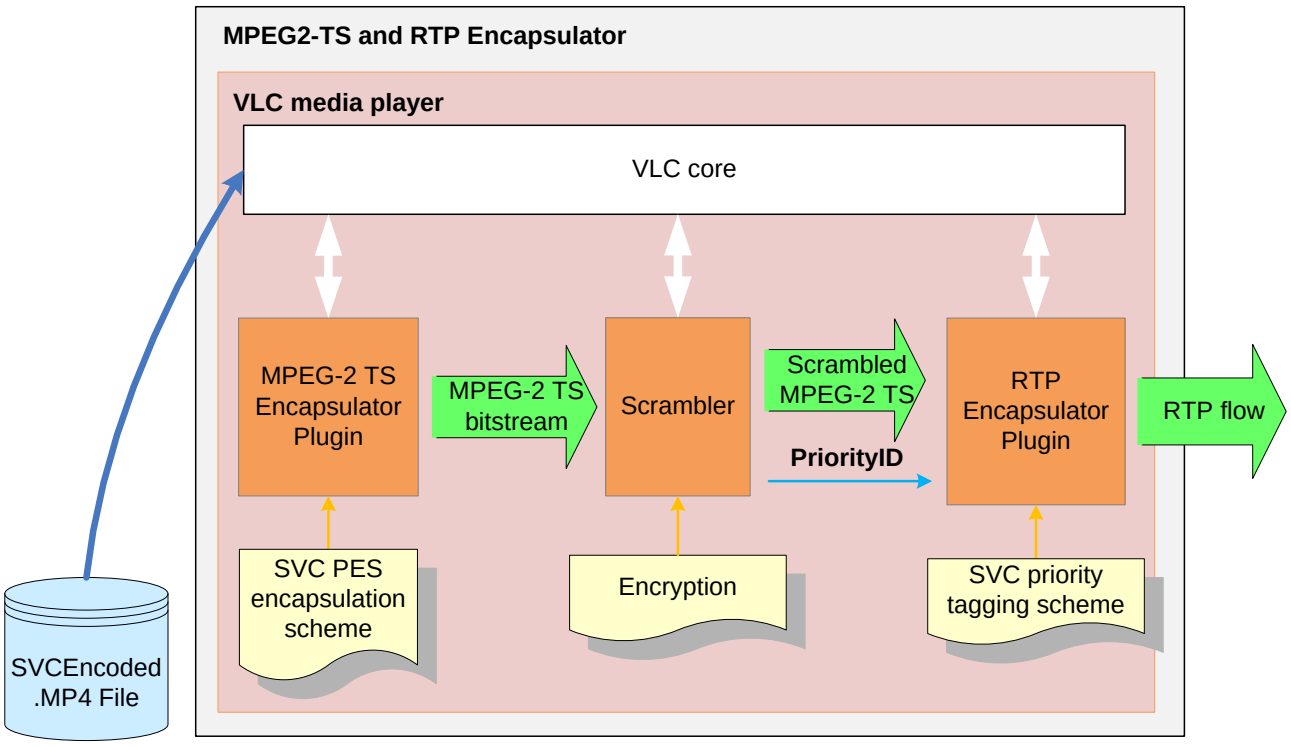


Figure 22 – SVC MPEG-2 TS and RTP Encapsulator as plug-ins for the VLC media player

The MPEG-2 TS Encapsulator plugin for SVC generates an MPEG-2 TS from a given MP4 file, in compliance with the SVC specific encapsulation scheme [19]. The SVC encapsulation approach for MPEG-2 TS is already presented in detail with corresponding diagrams in [4] (Figures 22-24). The generated MPEG-2 TS, which must have at least one NULL packet every 300ms to allow ECM packets insertion at the encryption process, is encrypted by the Scrambler, as shown in Figure 20. The scrambled MPEG-2 TS together with Priority ID (Priority ID is associated to the pair of {T (temporal layer), Q (quality layer)} provided

by the NAL unit header) is the input for the RTP Encapsulator plugin. The reason for doing encryption in this way and not by an external device is that the temporal level and quality level of a NAL unit is indicated in the NAL unit header and this is no longer accessible after encryption. Therefore, the encryption has to be done as proposed here and the Scrambler has to provide the Priority ID mapping to the RTP Encapsulator plugin. Only TS packets that contain NAL units with the same quality level and temporal level (i.e., same Priority ID) are sent within the same RTP packet. Thus, the Packet Dropping Device can either keep or drop a complete RTP packet. The metadata (i.e. Priority ID signalling) needed at the PDD for deciding whether to keep or drop an RTP packet is signalled using the SSRC field in the RTP header, as shown in Figure 1.

5.3 Encryption and decryption component

As aforementioned, it is no possible to use a standard standalone scrambler application with SVC-based approach, since the scrambling process would lead to a loss of the quality level and temporal level information and further it may scramble together TS packets containing NAL units with different temporal a quality levels. Therefore, for the prototype implementation of the SVC-based approach, the scrambling/descrambling functionality is implemented in the form of library that can be integrated in between MPEG-2 TS Encapsulator plugin and RTP Encapsulator plugin as shown in Figure 20. The interfaces of the library are given in the following two sections.

5.3.1 Scrambler (class)

In this section the main parts of the scrambler class are presented:

scrambler(void);

scrambler(unsigned __int32 bitrate);

~scrambler(void);

Constructors and destructor of the class. If calling the constructor with the bit rate parameter, it specifies bit rate of input MPEG-2 TS data stream that is used to calculate the ECM repetition time (which is by default once every 300 ms) and the CW period (which is by default 10 seconds). This value is based on bit rate of the full MPEG-2 TS data stream.

unsigned __int32 get_Bitrate(void);

void set_Bitrate(unsigned __int32 value);

Get and Set the bit rate of the input MPEG-2 TS data stream, the bit rate value is used to calculate the ECM repetition time (which is by default once every 300 ms) and the CW period (which is by default 10 seconds). This value is based on bit rate of the full MPEG-2 TS data stream.

unsigned __int16 get_PMTPID(void);

void set_PMTPID(unsigned __int16 value);

Get and Set the PMT PID, the scramble method will parse the incoming MPEG-2 TS data stream and will set the PMT PID after finding and successfully parsing a PAT table. By setting the PMT PID manually the initialization process can be made faster resulting in less clear content before scrambling can start.

unsigned __int16 get_ECMRepTime(void);

void set_ECMRepTime(unsigned __iEmcnt16 value);

Get and Set the ECM repetition time in milliseconds, default 300 msec.

unsigned __int16 get_CWPeriod(void);

void set_CWPeriod(unsigned __int16 value);

Get and Set the CW period in milliseconds, default 10000 msec.

*scrambler_PIDList *get_ComponentList(void);*

Get a list of components which are supposed to be scrambled.

*bool scramble(unsigned __int8 *mpegts, unsigned __int32 len);*

Process and if possible scramble a certain amount of mpeg packets. The given *len* must be a multiple of 188 bytes. This method extracts and parses the PAT and PMT to obtain the components of the given service. Once the components are known scrambling is started (any content packets prior to this remain in the clear). This method alters each occurrence of the PMT to add the CA Descriptor and scrambles each content packet with a random Control Word. At intervals an ECM is added to allow descrambling. The method returns TRUE if at least one packet from the input is scrambled.

*bool scramble(unsigned __int8 *mpegts, unsigned __int32 len, bool baseBandContent);*

Like above but only adds ECMs if boolean baseBandContent is set to TRUE.

5.3.2 Descrambler (class)

In this section the main parts of the descrambler are presented:

descrambler(void);

~descrambler(void);

Constructor and destructor of the class.

unsigned __int16 get_PMTPID(void);

void set_PMTPID(unsigned __int16 value);

Get and Set the PMT PID, the scramble method will parse the incoming MPEG-2 TS and will set the PMT PID after finding and successfully parsing a PAT table. By setting the PMT PID manually the initialization process can be made faster resulting in less clear content before scrambling can start.

*scrambler_PIDList *get_ComponentList(void);*

Get a list of components which are supposed to be scrambled.

*bool descramble(unsigned __int8 *mpegts, unsigned __int32 len);*

Process and if possible descramble an certain amount of mpeg packets. The given *len* must be a multiple of 188 bytes. This method extracts and parses the PAT and PMT to obtain the components of the given service and parses the PMT to obtain the ECM stream. Once ECM stream PID is known and the components PIDs are known the Control Words are used to descramble the content. This method returns TRUE if at least one packet from the input is descrambled.

5.4 SVC Set-Top-Box emulator

Figure 23 illustrates the SVC STB emulator. It is based also on the VLC media player implementation [8] and is used to decode the SVC bit stream. RTP Access and Decoder Plug-ins as well as the SVC Decoder have been developed by Fraunhofer HHI. The PC-based STB emulator that embeds the RTP Access and Decoder Plug-ins and provides a STB-like interface has been developed by UDC.

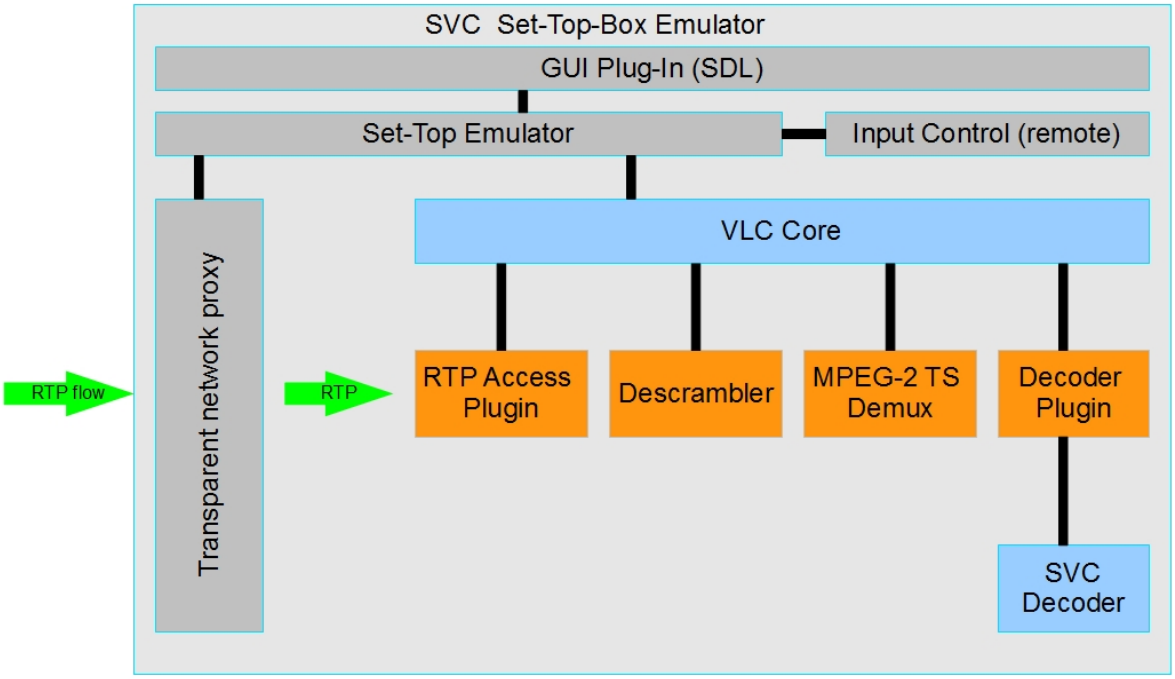


Figure 23 – SVC Set-Top-Box emulator

The STB emulator is fully implemented in software and running on a regular PC, with no hardware assistance for audio / video decoding; the lack of a specialized hardware unit is more than compensated by the massive difference in processing power over a real set-top box.

The emulator is a software engine that implements the same basic functionality as the embedded client in a real set-top box; currently it has support for retrieving video streams from UDP, switching channels under user control, displaying a graphical user interface, receiving input commands (key presses) from a remote control unit, and decoding and rendering the video.

Video decoding and rendering is performed by embedding a VLC player (VideoLan Client) running in slave mode, controlled by the set-top-box emulator. The emulator can start and then control (pause, play, switch media) VLC; it plays the same exact role as the demuxer / decoder chain in a physical set-top-box.

The RTP access and decoder plugins developed by HHI are dynamically loaded as modules by VLC. Without them, the set-top-box emulator does not work for SVC flows, but can be used for regular H.264 AVC content.

The Input Control module is implemented as a pluggable module in the emulator; currently it reads keyboard presses or buttons from a universal Gyration remote control. Alternate versions can be developed with ease to read an infrared port or a Bluetooth socket.

Similarly, we have tried to keep the Graphical User Interface module separate from the emulator engine; however it must be noticed that multiple aspects of the user interface in a set-top-box are alien to most graphics interfaces – there is no real concept of “windows”, but rather there are “planes”; the “focus” is somewhat less clear, etc.

Currently, the Graphical User Interface is developed using the SDL library.

Additionally, the emulator may start an optional network proxy; it is intended to monitor the incoming streams and provide some simple insight on the traffic. Currently it is simply monitoring instantaneous bit rate in a short sliding window (1 second), and a real-time bit rate graph can be superimposed on the screen by pressing a key.

All components have been chosen to be easily ported. While the usual platform at the UDC group is Linux, the RTP plugin and SVC decoder are usually developed for Windows. We have chosen libraries and components (VLC, SDL, Erlang/OTP) that can be simply ported to any of these, or other, platforms. Right now the set-top-box emulator runs both on Windows and Linux – of course, on Linux only AVC streams can be decoded.

5.5 Test environment for Head End prototype

Figure 24 presents a test environment for the SVC-based Head End prototype which has been developed and built by HHI and used for demonstrations of the SVC-based packet dropping solution. It is used to validate the MPEG-2 TS [17] and RTP [20] encapsulation of SVC encoded streams and additional metadata tagging (PriorityID in SSRC filed of the RTP header) at the SVC-based Head End. More concretely, it is tested that RTP packets can be dropped at the SVC-based Packet Dropping Device and the resulting content can be correctly played. This test environment consists of three main components:

a) Head End prototype which includes MPEG-2 TS and RTP Encapsulators
b) PC-based PDD & STB Emulator with the SVC Decoder. This PC is connected to an HD display.
c) Web Interface for SVC MGS scalability control

Figure 24 illustrates the general functionality of the SVC-based rate adaptation and packet dropping. The SVC Head End generates the SVC SPTS over RTP described in this document and it is sent to the PC connected to an HD display, where the PC-based PDD and the STB emulator are running (SVC PDD & STB Emulator in the Figure).

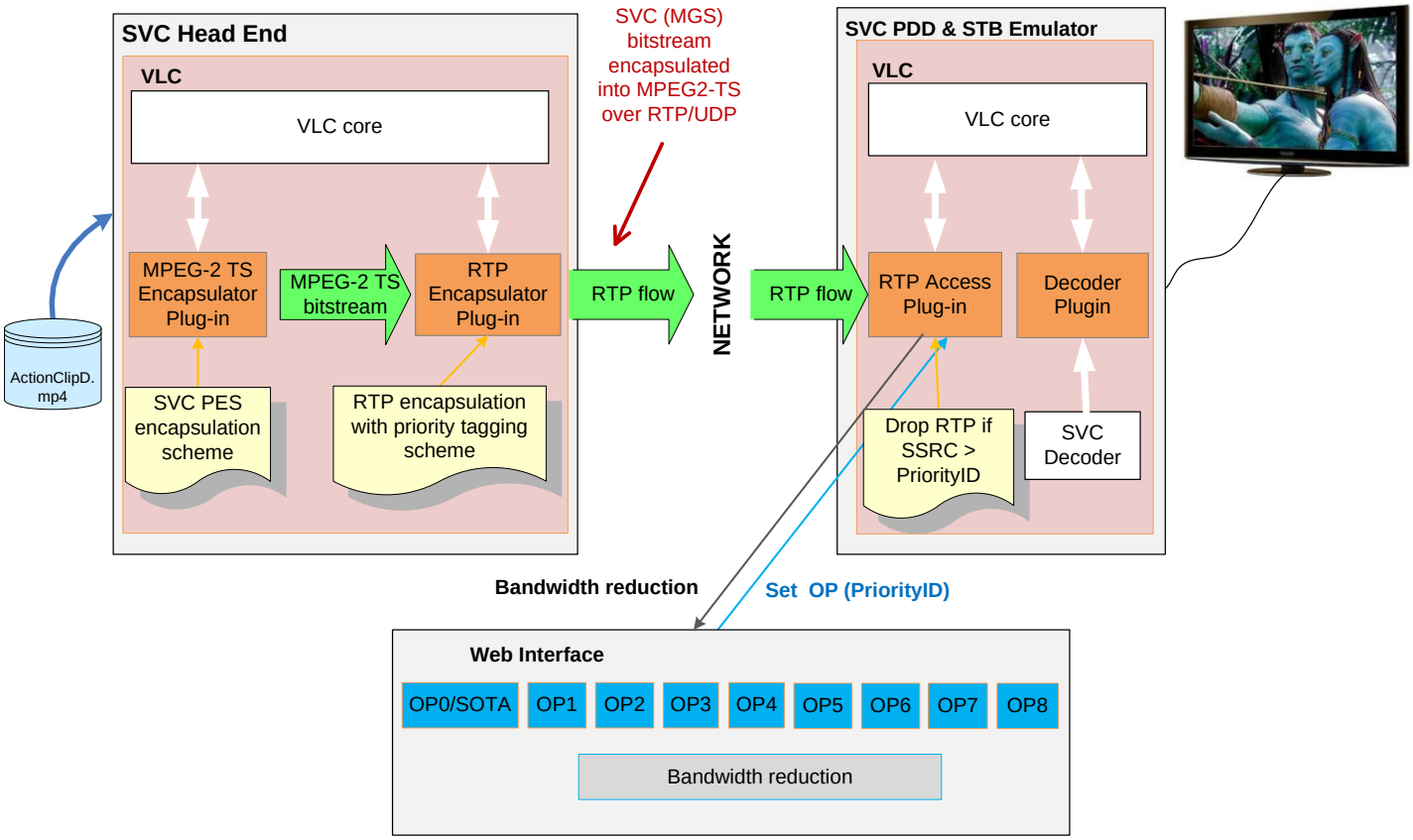


Figure 24 – SVC Head End prototype test

Figure 25 shows the screenshot of the Web Interface. Using this Web interface several Operation Points can be selected in real time. Information about a certain OP is provided to the RTP Access Plug-in of the SVC-based PDD & STB Emulator. It allows to test and emulate the simple logic of the SVC-based Packet Dropping Algorithm and the dropping of RTP packets as well as to evaluate the MPEG-2 TS and RTP

encapsulation process at the Head End. If a certain Operation Point has been selected on the Web Interface then information about the appropriate PriorityID is immediately sent to the RTP Access Plug-in. In this case, all RTP packets which have a Priority ID in their SSRC field higher than the one indicated by the Web Interface are dropped. The payload of remaining RTP packets will be used for decoding. The bandwidth reduction and the change of the visual video quality after RTP dropping can be tested in real-time.

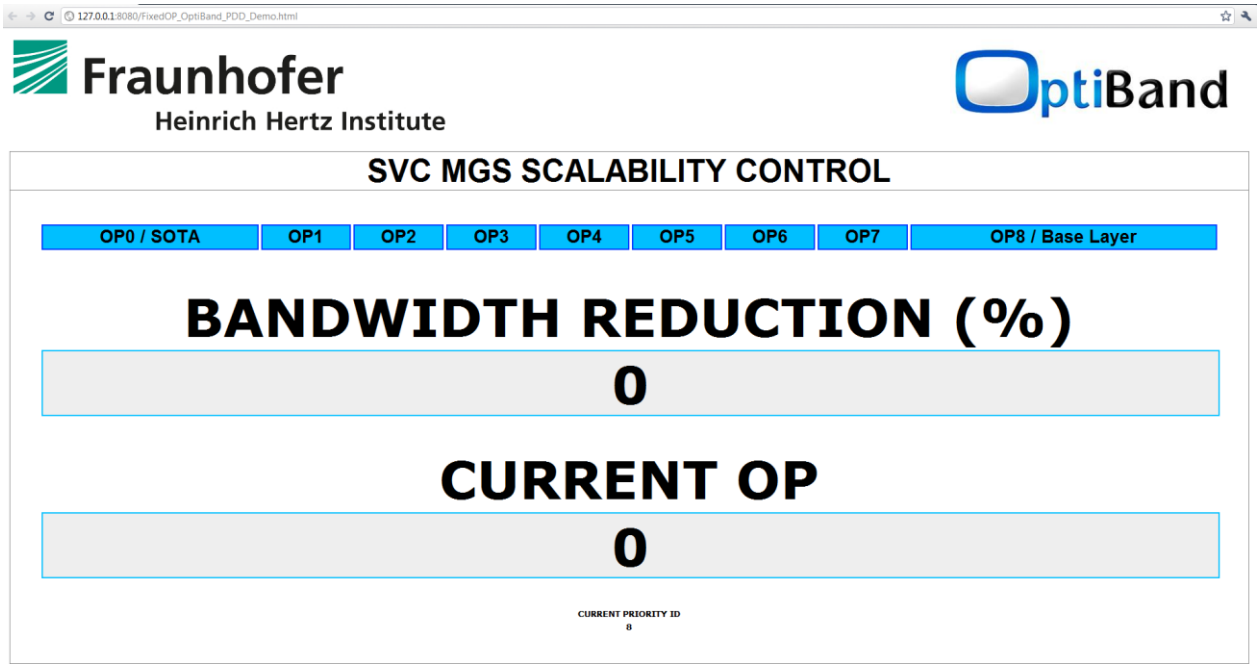


Figure 25 – Web Interface for SVC MGS scalability control

5.5.1 Head End prototype conformance check and system test

The Head End processing (encoding, content analysis, metadata tagging, encapsulation and transmission of the video content) has been validated in previous section verifying that the rate adaptation of HD video can be achieved at the Packet Dropping Device for the SVC-based approach. After checking the correct operation of the SVC Head End and SVC environment shown in Figure 24, results of the prototype are obtained where it is checked the objective of reaching 33% bandwidth reduction can be fulfilled. This section describes a conformance check and system test of the Head End prototype of the SVC-based approach for packet dropping.

The following video sequences "CC1_Action_ClipA", "CC1_Action_ClipB", "CC1_Action_ClipC" (max. rate ca. 8.2Mbps) are used for tests of Head End prototype (see D2.1 Criteria specification for the QoE research for more details). This video sequence is used in WP2 for subjective tests. It is targeted to a STB over ADSL rate of 5.5Mbps (with ca. 33% bandwidth reduction). The action movie clips "CC1_Action_ClipA", CC1_Action_ClipB", "CC1_Action_ClipC" are used as inputs to the Head End implementation.

RTP packets are tagged with the Priority ID for further analyzing at the PDD for packet dropping.

Three trace files have been generated during the Head End testing using the VLC tool with MPEG-2 TS Encapsulator and RTP Encapsulator plug-ins developed by HHI.

-ActionClipA_Optiband_IPTV_Head_End_Simulation_RTP_Encapsulation.trc

-ActionClipB_Optiband_IPTV_Head_End_Simulation_RTP_Encapsulation.trc

-ActionClipC_Optiband_IPTV_Head_End_Simulation_RTP_Encapsulation.trc

They represent the RTP encapsulated streams and are used for the corresponding simulation of the SVC-based Packet Dropping Devices in T3.3. These trace files contain the list of RTP packets of the video streams and indicate the parameters needed by the PDD simulator in order to adapt video streams.

A sample from a trace file (CC1_ActionClipA) generated at the Head End is shown in *Table 12*.

Table 12 – RTP Encapsulation with priority tagging

```
'RTP' PrioID, Seq#, Size, flag, PT, RTP timestamp
...
RTP 5, 5193, 1328, 0, 33, 3792096080;
RTP 5, 5194, 1140, 0, 33, 3792096148;
RTP 0, 5195, 1328, 0, 33, 3792096206;
RTP 0, 5196, 1328, 0, 33, 3792096274;
RTP 0, 5197, 1140, 0, 33, 3792096342;
RTP 2, 5198, 1328, 0, 33, 3792096401;
RTP 2, 5199, 1328, 0, 33, 3792096469;
RTP 2, 5200, 1328, 0, 33, 3792096537;
RTP 6, 5201, 1328, 0, 33, 3792096605;
RTP 6, 5202, 1328, 0, 33, 3792096673;
RTP 6, 5203, 1328, 0, 33, 3792096741;
RTP 6, 5204, 1328, 0, 33, 3792096809;
RTP 6, 5205, 1328, 0, 33, 3792096877;
RTP 6, 5206, 1328, 0, 33, 3792096946;
```

Following parameters are written into a trace file:

-PrioID The RTP packet is tagged with the Priority ID using the SSRC field of the RTP header. It is the most important metadata information for the PDD which will be generated at the Head End;

-Seq# Sequence number of the RTP packet which can be used to detect lost packets;

-Size Size of the RTP packet (measured in bytes);

-Flag M bit flag. Set to 1 whenever the timestamp is discontinuous;

-PT Payload type which is 33 in case of MPEG-2 Transport Stream packets according to RFC 2250;

-RTP timestamp 32 bit 90K Hz timestamp representing the target transmission time for the first byte of the packet;

Figure 26, Figure 27 and Figure 28 show the Head End output in terms of data rate variations over time for different Operation Points for the action sequences ActionClipA, ActionClipB and ActionClipC respectively for the second version of the Head End.

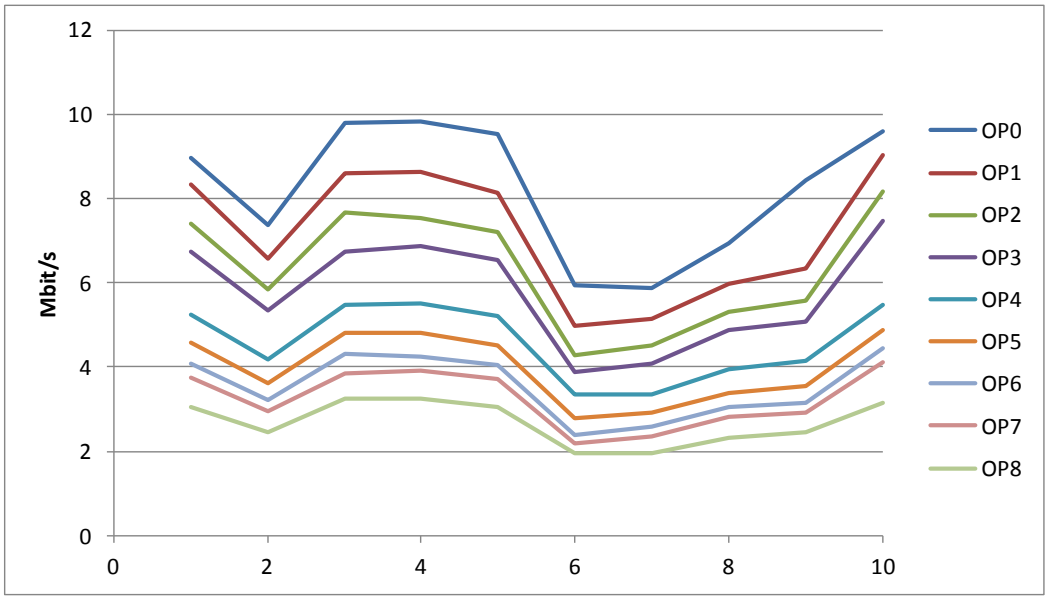


Figure 26 – Data rate for each Operation Point for ActionClipA movie (HD video sequence "Avatar" 10s long 1080p@25)

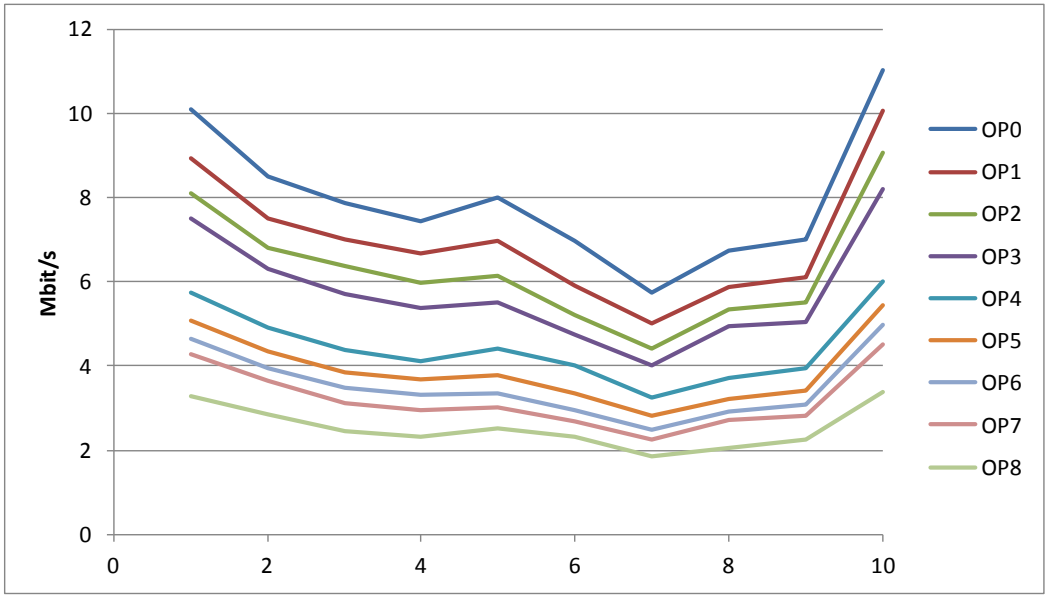


Figure 27 – Data rate for each Operation Point for ActionClipB movie (HD video sequence "Avatar" 10s long 1080p@25)

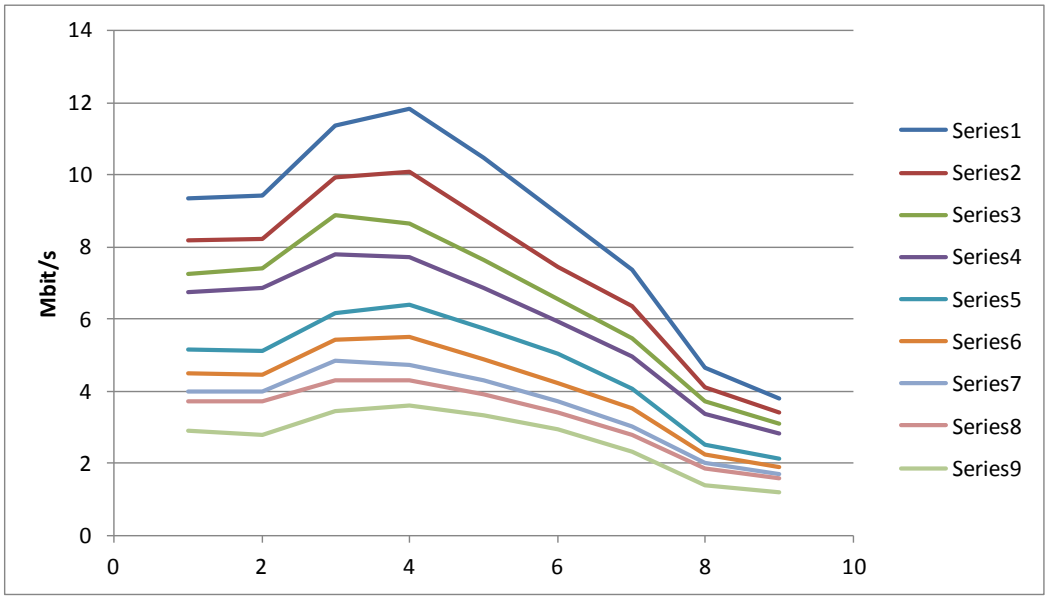


Figure 28 – Data rate for each Operation Point for ActionClipC movie (HD video sequence "Avatar" 10s long 1080p@25)

Table 13 shows rate properties of SVC encoded streams (VBR) with respect to different Operation Points. With the results presented in the figures and in the table, it can be appreciated that the objective of reaching a 33% bandwidth reduction can be obtained with the SVC stream. In fact, there transmitted operation point should be under 5.5Mbps and in these results it can be seen that there is always an OP which bandwidth is under this value. Since the SVC streams are encoded with VBR encoding switching from one operation point to another is performed over the time to adjust better to the available 5.5Mbps.

Table 13 - VBR SVC Streams - Rate Properties

	Avg. rate [kbps] OP0	Avg. rate [kbps] OP1	Avg. rate [kbps] OP2	Avg. rate [kbps] OP3	Avg. rate [kbps] OP4	Avg. rate [kbps] OP5	Avg. rate [kbps] OP6	Avg. rate [kbps] OP7	Avg. rate [kbps] OP8 /Base Layer
CC1_Action_ClipA	8225,3	7121,6	6295,9	5703,0	4559,6	3960,2	3521,5	3220,1	2670,7
CC1_Action_ClipB	7964,2	7024,9	6297,4	5708,4	4460,7	3909,9	3519,7	3185,9	2531,5
CC1_Action_ClipC	8476,3	7384,3	6515,6	5892,2	4697,6	4072,1	3590,2	3281,1	2652,5
CC1_Action_ClipD	8213,6	7349,7	6621,4	6011,9	4316,5	3853,0	3501,8	3190,7	2319,7

The test environment for the SVC-based Head End prototype described in section 5.5 has been used to verify the functionality of the prototype (the MPEG-2 TS and RTP encapsulation at the Head End) and to test the SVC-based Packet Dropping Algorithm. The dropping of RTP packets has been also emulated as well as a video quality after such a dropping process has been analyzed.

Besides the streams shown above used for verifying the functionality of the prototype, a "soccer" sequence and a "documentary" sequence have been generated to test the correct operation of the PDD [6] prototype and the interoperability with the STB. The bit rates over the time of these sequences are shown in the figures below for 10 seconds.

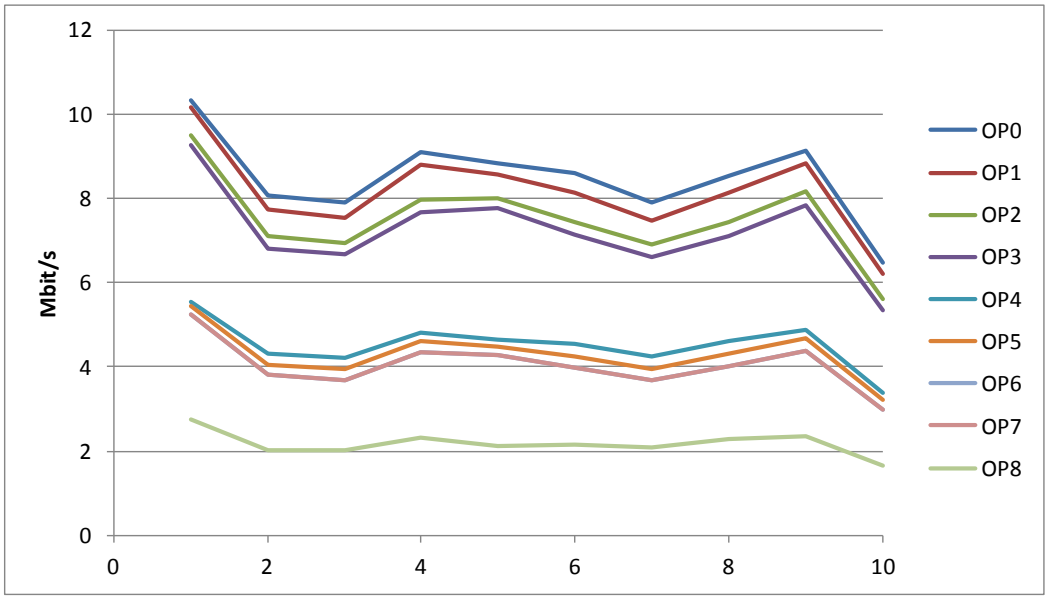


Figure 29: **Data rate for each Operation Point for Soccer movie (10s long 1080p@25)**

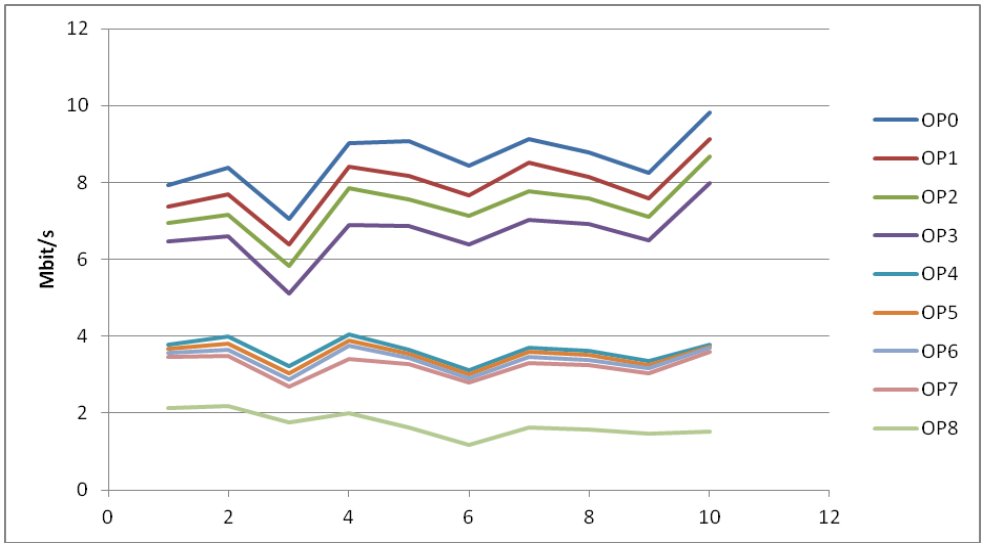


Figure 30: **Data rate for each Operation Point for Documentary movie (10s long 1080p@25)**

The look-up-table for the SVC approach is shown in **Table 14**. In the table the MOS values extracted from [7] are shown indicating also the OPs, which each of the bandwidth levels corresponds to. As can be seen the number of different *bandwidth_levels* is higher in the case of SVC as already mentioned before.

Table 14: Look-up-Table (LUT): MOS values for the bandwith_levels and corresponding operation points for SVC approach

Stream_type	bw_level_0 OP0	bw_level_1 OP1	bw_level_2 OP2	bw_level_3 OP3	bw_level_4 OP4	bw_level_5 OP8	bw_level_6 O9
Action	4,375	4,25	4,16	4,09	4,19	4,06	3,88
Soccer	3,854	3,82	3,86	3,73	3,67	3,59	3,72
Documentary	4,1395	4,06	4,035	3,935	3,955	3,85	3,825

6. Conclusions

The second and final version of the Head End prototype has been carried out for different tagging approaches at the Head End device with respect to the three approaches:

- AVC video encoding with multiple profiles,
- Scalable Video Coding with MGS SNR scalability

This document builds the technical basis of all technologies involved in the Head End. It includes the Head End prototype description, encryption components, middleware, STB, configuration of the network, interaction between Head End and Packet Dropping Device, used metadata information and achieved prototype results of the Head End processing.

7. Bibliography (optional)

[1] D1.1 - Functional specification document

[2] D3.2 - IPTV data dropping algorithm description

[3] D3.4.1 - IPTV data dropping algorithm prototype Version 1 and associated documentation

[4] D4.1 - Head end video processing description

[5] D4.3 - Metadata objects description

[6] D3.4.2 - IPTV data dropping algorithm prototype Version 2 and associated documentation

[7] D2.4 - Intermediate QoE research recommendations report

[8] VideoLAN, available at “http://www.videolan.org”.

[9] H. Schwarz, D. Marpe, and T. Wiegand: “Overview of the Scalable Video Coding Extension of the H.264/AVC Standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, Special Issue on Scalable Video Coding, Vol. 17, No. 9, pp. 1103-1120, September 2007.

[10] Thomas Wiegand, Ludovic Noblet, and Fabrizio Rovati: “Scalable Video Coding for IPTV Services”, *IEEE TRANSACTIONS ON BROADCASTING, VOL. 55, NO. 2, JUNE 2009*.

[11] Ingo Kofler, Robert Kuschnig and Hermann Hellwagner: “Improving IPTV Services by H.264/SVC Adaptation and Traffic Control”, Proceedings of the IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB 2009), May 2009.

[12] S. Wenger, Y.-K. Wang, T. Schierl, and A. Eleftheriadis, “RTP Payload Format for SVC Video.” Internet Draft, Nov. 2008

[13] Mylene C. Q. Farias, "Video Quality Metrics", Digital Video, Book edited by: Floriano De Rango, ISBN 978-953-7619-70-1, pp. 500, February 2010, INTECH, Croatia.

[14] Broadcom, System-on-Chip solution for Full-Resolution HD 3DTV supporting SVC (and MVC) http://www.broadcom.com/products/Satellite/Satellite-Set-Top-Box-Solutions/BCM7425 , (accessed April 29, 2011)

[15] SVC Reference Software (JSVM software), http://ip.hhi.de/imagecom_G1/savce/downloads/SVC-Reference-Software.htm, (accessed Mai 9, 2011)

[16] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, “Overview of the H.264/AVC video coding standard,” *IEEE Trans. Circuits Syst.Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[17] ITU-T Rec. H.222.0, “Information Technology — Generic Coding of Moving Pictures and Associated Audio Information: Systems,” May 2006

[18] ISO/IEC 13818-1:2007, “Information Technology — Generic Coding of Moving Pictures and Associated Audio Information (MPEG-2) —Part 1: Systems,” 2007.

[19] **ISO/IEC 13818-1:2007/FDAM 3 - "Transport of Scalable Video over ITU-T Rec H.222.0 | ISO/IEC 13818-1"**

[20] RTP Payload Format for MPEG1/MPEG2 Video, IETF, RFC 2250, http://tools.ietf.org/html/rfc2250

[21] Annex I – “Description of Work”