Understanding Timelines within MPEG Standards

Lourdes Beloqui Yuste, *Member, IEEE,* Fernando Boronat, *Senior Member, IEEE,*
Mario Montagud, *Member, IEEE,* and Hugh Melvin, *Member, IEEE*

***Abstract*— Nowadays, media content can be delivered via diverse broadband and broadcast technologies. Although these different technologies have somehow become rivals, their coordinated usage and convergence, by leveraging of their strengths and complementary characteristics, can bring many benefits to both operators and customers. For example, broadcast TV content can be augmented by on-demand broadband media content to provide enriched and personalized services, such as multi-view TV, audio language selection and inclusion of real-time web feeds. A piece of evidence is the recent Hybrid Broadcast Broadband TV (HbbTV) standard, which aims at harmonizing the delivery and consumption of (hybrid) broadcast and broadband TV content.**

A key challenge in these emerging scenarios is the synchronization between the involved media streams, which can be originated by the same or different sources, and delivered via the same or different technologies. To enable synchronized (hybrid) media delivery services, some mechanisms providing timelines at the source side are necessary to accurately time align the involved media streams at the receiver-side. This paper provides a comprehensive review of how clock references (timing) and timestamps (time) are conveyed and interpreted when using the most widespread delivery technologies, such as DVB, RTP/RTCP and MPEG standards (e.g., MPEG-2, MPEG-4, MPEG-DASH and MMT). It is particularly focused on the format, resolution, frequency and the position within the bitstream of the fields conveying timing information, as well as on the involved components and packetization aspects. Finally, it provides a survey of proofs of concepts making use of these synchronization related mechanisms.

This complete and thorough source of information can be very useful for scholars and practitioners interested in media services with synchronization demands.

***Index Terms*—Media Synchronization, Timelines (Clock References and Timestamps), MPEG, ISO BMFF, MPEG-DASH, MMT, RTP, RTCP.**

I. Introduction

At present, there is a huge variety of technologies to deliver time-sensitive media content in networked environments [1]. On the one hand, broadcast technologies, such as DVB (Digital Video Broadcasting), can concurrently deliver the same content to a large number of users. In this context, media can be broadcasted by using terrestrial (e.g., DVB-T), satellite (e.g., DVB-S), mobile (e.g., DVB-H), and cable (e.g., DVB-C) technologies. On the other hand, the unceasing advances in (IP) broadband delivery technologies, combined with their widespread deployment, has sparked the growth in media delivery using this kind of distribution channels [2]. In this context, media can be delivered by using different forms of streaming and downloading techniques.

For broadcast delivery, Moving Picture Experts Group (MPEG) standards are the means used by DVB technologies. For broadband delivery, Real-Time Transport Protocol (RTP) [3] and HTTP-based Adaptive Streaming (HAS) solutions are commonly used [1]. For instance, MPEG has proposed Dynamic Adaptive Streaming over HTTP (MPEG-DASH) [4], as a client-driven streaming solution that aims at improving the adaptability, smoothness and continuity of media play-out under variable network conditions.

In general, broadband delivery technologies provide poorer performance than broadcast delivery technologies in terms of scalability, stability, and latency. However, a clear benefit of using broadband delivery methods is the availability of bidirectional communication capabilities, unlike the one-way nature of broadcast delivery methods. This enables the development of interactive and customized media services through users' requests.

Although broadcast and broadband delivery technologies have somehow become rivals in the competitive media consumption market, the inter-operability, coordination and seamless convergence between both, by leveraging their strengths and complementary characteristics, can offer a lot of new possibilities, opening the door for new business models. This is particularly relevant to TV operators and other stakeholders (e.g., device manufacturers, content providers...), since the (linear) broadcasted TV content can be augmented by on-demand media content delivered via broadband networks to provide enriched media services. Various examples are ([5], [6], [7]): provision of free viewpoint TV, high definition media, tiled streaming (e.g., ultra high resolution video distribution where different spatial areas of the same video are delivered as different streams), concurrent consumption of various video streams (either picture-in-picture, in a mosaic view or in different devices) or switching between them, customized selection of audio streams, (targeted) commercials, integration of web feeds (e.g., widgets, quizzes, statistics...) and Social Media, etc. This enriched media consumption paradigm is not only targeted for entertainment purposes, but it can also bring social benefits, such that users can feel more integrated and immersed when consuming media. Examples are native audio language selection, inclusion of videos with sign language or adapted subtitles for people with audio/visual impairments, etc.

As a typical scenario, let us consider the broadcasting of a sport event. In such a case, fans living abroad may want

Lourdes Beloqui Yuste received in 2015 her PhD in Information Technology from the College of Engineering & Informatics, National University of Ireland, Galway. lbeloqui@gmail.com.

Dr. Fernando Boronat is a lecturer at the Universitat Politècnica de València, Campus de Gandia. València, Spain. fboronat@dcom.upv.es.

Dr. Mario Montagud is a Postdoc researcher at the University Politècnica de València, Campus de Gandia. València, Spain. In 2015, he is doing a PostDoc at Centrum Wiskunde & Informatin (CWI), Amsterdam, the Netherlands mamontor@posgrado.upv.es montagud@cwi.nl.

Dr. Hugh Melvin is a lecturer at the College of Engineering & Informatics. National University of Ireland, Galway. hugh.melvin@nuigalway.ie.

Manuscript received February xx, 2015; revised October xx, 2015.

to watch their home team playing an important game, but are forced to listen to a local company's (perhaps biased) embedded audio commentary. It would be especially interesting to have the chance of substituting this built-in audio stream by the one from their favourite (home local) radio station (e.g., provided via Internet). Another desired feature may be to simultaneously watch the TV signals from different operators or content providers. As an example, this would allow viewers to be aware of the reactions of the opposite teams fans to specific actions and to experience with different perspectives of the game. Furthermore, the inclusion of web feeds would allow the viewers to receive real-time notifications about news, statistics or contextual information.

All the above use cases require additional bandwidth, which is a scarce and expensive asset in the broadcast world. Accordingly, the enrichment of broadcast media services with additional, but related, broadband media services becomes an optimal approach, as it also provides flexibility for requesting or not the particular additional content, depending on users' interests, needs or profiles.

Due to the potential of the convergence between broadcast and broadband technologies, ongoing standardization activities were recently started in this area. On the one hand, Hybrid Broadcast Broadband TV (HbbTV)[1] [8] standard aims at harmonizing the delivery of interactive broadcast and broadband TV services through connected TVs, set-top boxes and multi-screen devices. It specifies signalling aspects, details the mechanisms to access and deliver the available media, and provides basic techniques for a concurrent presentation of the related media streams. On the other hand, the MPEG Media Transport (MMT) standard [9] (explained later) also focuses on the delivery of diverse types of media content over heterogeneous networks, which can be accessed anywhere from a large variety of devices.

This complex media ecosystem, in which a large variety of media types can reach diverse types of consumption devices using various encoding mechanisms, delivery protocols, and networks, faces many challenges. Even though ongoing research efforts are working towards a seamless integration and inter-operability between the available delivery technologies, the potential of hybrid media delivery is still not fully exploited. In particular, a key technological aspect that still needs further research is the ability to accurately synchronize the presentation of all involved media streams.

Indeed, recent studies have shown that the magnitudes of end-to-end delay differences when streaming media content via different delivery technologies are much larger than acceptable limits ([10], [11]), thus revealing the need of synchronization (sync hereafter) between streams.

A fundamental requirement to enable synchronized (hybrid) media delivery services, consist of the availability of a coherent framework for precisely inserting, interpreting and aligning timelines (i.e., timing information) into the delivered media through the end-to-end distribution chain. This is essential for reconstructing the original timing of the individual incoming media streams at the receiver-side (especially relevant in

[1] www.hbbtv.org

TABLE I: Abbreviations

Acronyms	
AAC	Advanced Audio Coding
ADC	Asset Delivery Characteristics
AF	Adaptation Field
AU	Access Unit
AVC	Advanced Video Coding
BIFS	Binary Format for Scenes
CA	Clock Accuracy
CI	Composition Information
CL	MPEG-4 Compression Layer
CoD	Content on Demand
CRI	Clock Relation Information
CTS	Composition Timestamps
ctts	Composition time-to-sample Box
CU	Composition Unit
DL	MPEG-4 Delivery Layer
DSM-CC	Digital Storage Media Command and Control
DTS	Decoding Timestamp
DTV	Digital TV
DVB	Digital Video Broadcasting
DVB SI	DVB Service Information
EIT	Event Information Table
ES	Elementary Stream
ESCR	Elementary Stream Clock Reference
FCR	FlexMux Clock Reference
GOP	Group of Pictures
HAS	HTTP Adaptive Streaming
HbbTV	Hybrid Broadcast Broadband TV Standard
HDS	HTTP Dynamic Streaming
HEVC	High Efficiency Video Coding
HE-AAC	High Efficiency Advanced Audio Coding
HLS	HTTP Live Streaming
IDES	Inter-Device Synchronization
IDMS	Inter-Destination Media Synchronization
IPTV	Internet Protocol TV
mdhd	Media Header Box
MDU	Media Data Units
MFU	Media Fragment Unit
MJD	Modified Julian Date
MMT	MPEG Media Transport
MMT DL	MMT Delivery Layer
MMT EL	MMT Encapsulation Layer
MMT SL	MMT Signalling Layer
MP2P	MPEG-2 Program Stream
MP2T	MPEG-2 Transport Stream
MP3	MPEG-2 Audio Layer 3
MP4	MPEG-4 part 14 MP4 File Format
MPEG	Moving Picture Experts Group
MPEG-2 PSI	MPEG-2 Program-Specific Information
MPEG-DASH	MPEG Dynamic Adaptive Streaming over HTTP
MPU	Media Processing Unit
MS-SSTR	Microsoft Smooth Streaming Protocol
MVC	Multi-view Video Coding
mvhd	Movie Header Box
NGN	Next Generation Networks
NTP	Network Time Protocol
OCR	Object Clock Reference
OD	Object Descriptor
OPCR	Original Program Clock Reference
OTB	Object Time Base
PAT	Program Association Table
PCR	Program Clock Reference
PES	Packetized Elementary Stream
PID	Packet Identifier
PLL	Phase-Locked Loop
PMT	Program Map Table
PS	Program Stream
PTP	Precision Time Protocol
PTS	Presentation Timestamp
PU	Presentation Unit
QoE	Quality of Experience
QoS	Quality of Service
RTCP	Real-Time Control Protocol
RTMP	Real-Time Messaging Protocol
RTP	Real-Time Protocol
SCD	System Clock Descriptor
SCF	System Clock Frequency
SCR	System Clock Reference
SCV	Scalable Video Coding
SDT	Service Description Table
SHVC	Scalable HEVC
SIDX	Segment Index
SL	MPEG-4 Sync Layer
SNTP	Simple Network Time Protocol
STB	System Time Base
stbl	Simple Table Atom Box
STC	System Time Clock
STD	System Target Decoder
stts	Decoding time-to-sample Box
TDT	Time and Date Table
tkhd	Track Header Box
TOT	Time Offset Table
TS	Transport Stream
T-STD	Transport Stream System Target Decoder
TVA	TV Anytime
UHDTV	Ultra High Definition TV
UTC	Coordinated Universal Time
VCO	Voltage Controlled Oscillator
VO	Video Object
VoIP	Voice over IP
VOP	Video Object Plane

packet-switched networks because of the delay variability), as well as for concurrently aligning the play-out for the related media streams in the time domain.

If multiple media streams from multiple sources need to be synchronized within specific receivers, it is necessary to comprehensively understand how all media delivery standards internally deal with timelines (clock references and time-stamps) to reproduce (i.e., time-align) encoder/source media clock to the decoder/receiver media clock. Accordingly, this paper provides a comprehensive review of how clock references (timing) and timestamps (time), are inserted/conveyed within the MPEG standards (particularly, MPEG-1, MPEG-2, MPEG-4, ISO BMFF, MPEG-DASH and MMT), RTP/RTCP protocols and DVB standards, which are the widespread solutions to deliver time-sensitive media content in current broadband and broadcast networks. This analysis is mostly focusing on the format, resolution, frequency and the position within the bitstream of the previously mentioned time-related fields in each one of the delivery technologies. Our goal is not to include a complete description of all the standards (readers can refer to the particular specifications for that), but rather to provide a solid and global source of information, with an exhaustive analysis of the involved components, the packetization aspects, and the essential fields that include such temporal information. We believe this paper will be very useful for any researchers and developers interested in distributed media systems with sync demands.

Other key aspects for media sync, such as clock sync, monitoring algorithms for delay differences calculation, and play-out adjustment techniques, are out of the scope of this paper.

The remainder of this paper is structured as follows. Section II describes some background to help understanding this paper. Section III details time/timing in MPEG Standards, while Section IV describes the DVB timelines. In Section V, the time mechanisms in RTP/RTCP protocol are explained. After that, Section VI compiles several proof of concepts that have made use of the above mechanisms to enable synchronized media services. Finally, Section VII concludes the paper with a summary and discussion. Table I lists the acronyms used in the paper.

II. Background

In order to help understanding this paper, a categorization of media sync types, some key concepts about clock issues, and a summary of delivery methods and standards, are introduced in this section.

A. Clock Issues

Two key concepts regarding clock references must be distinguished: global/absolute clock (wall-clock) and local/relative clock. Absolute clock time refers to a global time scale, such as Coordinated Universal Time (UTC), and is generally provided by an external clock source, such as by Network Time Protocol (NTP) [12] servers, Precision Time Protocol (PTP) servers or Global Positioning System (GPS). Local clock time is obtained through internal system (hardware or software) clocks, which can be, for example, provided by capturing devices. Local clocks may or may not be related to global time, and are typically used to reproduce the rate of advancement (ticks per unit of time) of encoder's and decoder's clocks. The use of relative timelines is common in multimedia systems with sync demands, while absolute (wall-clock references) are in some cases not strictly necessary for media sync purposes [13].

Clocks are typically used for three main purposes [14]: i) to specify the time of the day; ii) to arrange events' order; and iii) to measure time differences between events. The sync between the involved clocks in a media session can be essential for a good media sync performance. Even in the case of having initially synchronized the involved clocks, they will drift over time and, therefore, they need also to be periodically re-synchronized throughout the duration of the media session. The clock deviation parameters that can cause asynchrony situations are given in Table II. Clock parameters, such as resolution, offset, skew and drift are described in [15] and [16]. Clock resolution is '*the smallest unit by which the clock's time is updated. It gives a lower bound on the clock's uncertainty*' [16]. Thus, the resolution of a clock is an indicator of its granularity. Although the term resolution is generally used to characterize the physical clock, it is sometimes used interchangeably with the term precision, which is generally used to indicate the accuracy in reading the physical clock's resolution. For example, Microsoft's Windows 7 OS offers a precision of 15.625ms [17], irrespective of the physical clock's resolution, and the current version of Minix Operative System (OS) offers a precision of 16ms [18]. Around 2004, although typical clock's resolution was around 10ms, the tendency was to improve systems clock's resolution in various OS, such as Linux, FreeBSD, DragonFlyBSD, up to 1ms [19]. Clock frequency is the rate at which a physical clock's oscillator fluctuates. Thus, it represents the rate of change of that clock's time-scale with respect of true time.

Regarding clock skew (see definition in Table II), the example from [16] highlights the problem when measuring every minute one-way delays between two Internet hosts. For a transcontinental path, the transmission delay between the hosts could possibly reach up to 50ms. If the skew between the two clocks is 0.01% (i.e., 1 part in 10,000), then, in 10 minutes time frame, the accumulated error in the delay measurement is 60ms, which exceeds the transmission delay. Clock skews have a similar impact on media sync, as can be seen in Fig. 1, in which the audio/video asynchrony continuously increases due to this factor.

In [20], a solution for clock skew detection and compensation by using NTP and RTP/RTCP for Voice over IP (VoIP) applications is described. In particular, this method is based on the prerequisite that all system clocks are synchronized via NTP. The study of the RTCP Sender Reports (RTCP SR) packets analysing the increment of RTP timestamps and the NTP values indicates the presence or absence of skew. Skew between audio and system clock is present if the increment in both fields is not equal.

Another key issue is the distinction between time and timing. On the one hand, timing refers to the media clock's resolution/frequency. On the other hand, time or time-of-day

TABLE II: Parameters affecting Temporal Relationships within a Stream or among Multimedia Streams [13]

Cause	Parameter	Definition	Caused by
Network	**Network Delay**	Delay one packet experiences from the source, through the network, to the receiver.	Network load/traffic (congestion), network devices latency, serialization delay.
	Network Jitter	Delay variability.	Variable network conditions (e.g., load, traffic, congestion...).
End-System	**End-system jitter**	Delay at the end-systems caused by the task of packetizing/depacketizing AUs through protocols in different layers, encoding/decoding media, Operative System (OS) applications, jitter buffers, display lag, etc.	System load/hardware.
Clock	**Clock offset**	'*Difference in clock times*' [15].	Initialisation offset.
	Clock skew	'*First derivative of the difference in clock times*' [15]. Frequency difference.	Imperfections in clock manufacturing process.
	Clock drift	'*Second derivative of the difference in clock times*' [15]. Frequency change over time.	Temperature, pressure, voltage, crystal ageing, effect over time causing clock drift.

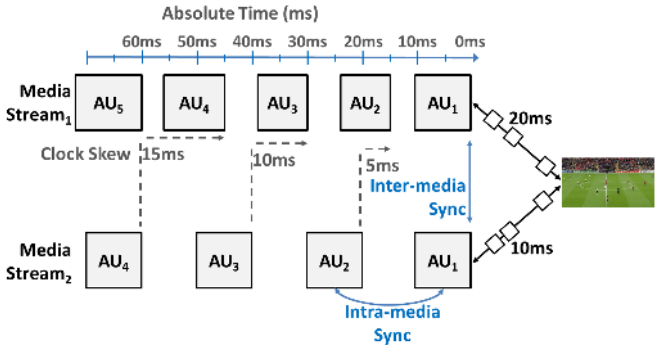


Fig. 1: Sync between two media streams (conveying two different media types). Figure shows Media Stream_1 with variable time length AUs and Media Stream_2 with a constant time length AUs

refers to a specific point in time denoted by some accepted time standard, such as UTC.

All the above clock factors are very relevant on media sync. Accordingly, in this paper we detail how timing clock references and timestamps information are inserted/conveyed in MPEG, RTP/RTCP and DVB standards.

B. Media Sync

Media streams are composed of Access Units (AU), which are also referred to as Media Data Units (MDU), within the various MPEG standards. An AU is the smallest timestamped media unit in a specific media stream. For example, a video AU is an encoded picture, whereas an audio AU is a set of encoded audio samples [21].

At the server side, the AUs of each particular media stream are captured, encoded and timestamped at particular (clock) rates. Thereafter, the AUs are packetised for transmission over the network. At the receiver-side, the de-packetisation, decoding and rendering processes must enable a reconstruction of the original temporal dependences between the AUs within and between the involved media streams. This end-to-end process for ensuring a proper and smooth media play-out is known as media sync.

However, multiple factors can have a significant impact on the media sync performance, especially when delivering media over packet-switched IP networks [22]. These factors (Table II lists the most relevant ones) can be categorized depending on whether they are associated with the network transmission, end-systems processes or end-systems clocks, and can be located at the server-side, network and/or receiver-side. For instance, (network) congestion results in packet loss, delay and jitter, while end-system processing load result in end-system delay. Likewise, clock oscillator characteristics, such as skew and drift, have an impact on the timing properties of the media streams. In this context, the work in [23] discusses the impact of several factors on audio-video sync, such as the acquisition equipment (e.g., microphones and cameras), programme composition (programme content), production equipment and processing, play-out equipment (e.g., audio and image output devices), the user's perception processing (spontaneous and cognitive user's response), as well as their relevance to the user's perceived Quality of Experience (QoE).

To illustrate the task of sync, one might consider a scenario where two individuals arrange to meet at a particular point in time. To succeed on this, three requirements must be met. First, they must agree to meet at a particular location at a particular point in time. Second, their clocks (i.e., time references) must operate at the same rate/frequency. Third, the individuals must have a common reference (or initial) point for their base times, such that no offset between their clocks exists. If this latter requirement is not met, then the individuals will arrive to the agreed location, but at different points in time.

Different types of media sync techniques can be distinguished. First, intra-media sync is needed to maintain the original temporal relationships between the AUs within each particular media type. Second, inter-media sync is required to preserve the temporal dependences between associated media types. Two main approaches can be followed when several media types are involved in a media application (e.g., audio, video, data...). The first one is to multiplex the individual media types into an aggregated stream, whilst the second one consists of independently transmitting each media type in a separate stream.

Fig. 1 illustrates the distinction between intra-media and inter-media sync. The former focuses on individual media types separately, while the latter involves multiple independent, but (semantically, spatially and/or temporally) re-

lated media types simultaneously (typically sent in different streams).

A specific sub-type of inter-media sync is referred to as inter-sender sync, which aims to synchronize the play-out of several media streams originated from different senders. It can also be possible that the media streams are delivered using different protocols, or even via different (e.g., broadcast and/or broadband) networks. In the latter case, this is usually referred to as hybrid sync. In specific cases, the different media streams can be played out on separate devices in a synchronized manner. This is usually known as inter-device sync or IDES (e.g., multi-screen applications).

There is an additional type of sync, named Point Sync, which requires the alignment of AUs at two sync points, which correspond to the beginning and the end of the display time [24]. For example, it is used for subtitles, which have an initial and final timestamp attribute associated with them. Together, these timestamps specify the period of time during with the subtitles should be presented to the viewer.

Apart from the above techniques that mostly aim to synchronize the play-out of different media streams within single devices (except for IDES), the simultaneous sync of the media play-out of specific streams across different devices is also needed. This is usually known as inter-destination media sync (or IDMS)[2].

As an example, audio/video sync (i.e., lip-sync) is the most characteristic case of inter-media sync. Several studies have conducted subjective testing to find out noticeable (or tolerable) asynchrony limits regarding lip-sync [23] [24] [25] [26]. In [24], it was shown that humans are more sensitive to audio leading (audio ahead of image) than audio lagging (audio behind image). In that work, the thresholds for lip-sync are divided into three ranges: undetectability (-95ms to +25ms), detectability (-125 to +45ms) and acceptability (-185 to +90ms). These asynchrony thresholds are shown in Fig. 2, in which the red area represents audio lagging/leading and the green area represents the user's undetectability ranges[3]. Tighter constraints are given in [26], where the acceptable asynchrony limits are bounded between +30ms in audio leading and -60ms in audio lagging [27]. Likewise, it is pointed out in [25] that a skew between -80ms (audio behind video) and +80ms (audio ahead of video) is noticeable, but tolerable for most users, whereas asynchrony levels exceeding -240ms or +160ms are intolerable. In that work, different Quality of Service (QoS) sync levels are also categorized, depending on the media, mode and application, ranging from tightly coupled audio/audio sync ($\pm 11\mu$s) to audio/pointer sync (-500ms to 750ms).

Regarding IDES, several allowable thresholds are given in [28]: $\pm 10\mu$s for tightly coupled audio; 2ms for real-time audio; 15ms for audio leading and 45ms for audio lagging in lip-sync; and $\pm$80ms for video animation.

For hybrid sync, it is also clear that different allowable asynchrony limits exist, depending on the specific use case, ranging from highly precise sync (e.g., frame accurate sync for TV mosaic views or multi-channel audio systems) to more relaxed requirements (e.g., subtitles or web feeds sync).

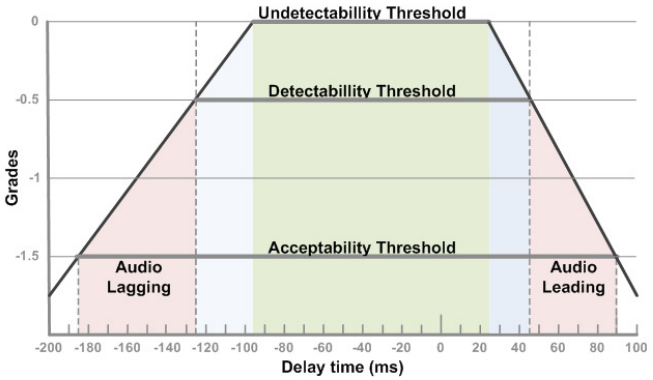


Fig. 2: Undetectability, detectability and acceptability threshold for lip-sync [24]

A large number of IDMS use cases can be found in [22], which are qualitatively ranked according to their sync requirements. The sync levels are: very high (10μs to 10ms); high (10-100ms); medium (100-500ms); and low (500-2000ms). For instance, networked stereo loud speakers require very high level sync; multi-party multimedia conferencing demands high level sync; second screen sync needs medium level sync; and finally, Social TV (which is the term to refer to social and community interaction using social networks, such as Facebook, while watching TV) requires low level sync.

C. Delivery Methods

Two main approaches for media delivery can be distinguished: broadcast and broadband [1]. Broadcast refers to the simultaneous delivery of media to all the users. In this paper we focus on the DVB standards, which differ in the employed physical platform: cable, DVB-C/C2 (ETSI EN 302 769); satellite, DVB-S/S2 (ETSI EN 302 307); terrestrial, DVB-T/T2 (ETSI EN 302 755), and hand-held (ETSI EN 302 304). Broadband technologies use IP networks as the delivery platform. In such a case, content can be delivered via unicast or multicast.

The broadband delivery methods are influenced by the IP network environment being used. In this context, two main forms of media streaming can be distinguished: managed and unmanaged [1] [2] [29] [30]. Managed services, such as cable TV or IPTV, are quoted services that operate within privately owned walled-garden IP environments. These services mainly rely on push-based multicast RTP/RTCP over UDP streaming, by using (semi-) professional *stateful*[4] servers, and provide service-compliant media delivery, including protection, authentication and re-transmission mechanisms. Contrarily, unmanaged or over-the-top (Internet) services, such as *WebTV*© or *TV on the Web*, are free services that can operate worldwide, and mainly employ *pull-based* unicast HTTP over

[2]The term IDES is also commonly referred to as special IDMS use case, in which the involved destinations are close to each other (e.g., different TV in a home), as the devices can also be considered destinations.

[3]A grade (y-axis) is a constant difference between detectable and acceptable thresholds (45ms for audio leading and 60ms for audio lagging).

[4]Server that retains state information about client's request

TABLE III: Media Delivery Techniques

Method	Applicability	File Download	Protocols	Drawbacks	Benefits
Downloading	Multiple use	Before play-out	HTTP/TCP	Waiting Time	No interrupted play-out
			IP Unicast	Bandwidth waste	No buffer is needed
Progressive Downloading	Web-based TV	During play-out	HTTP/TCP	Browser compatibility issues may exist	Reduced waiting time
			IP Unicast	Plugins for the play-out	
Streaming	IPTV	Along with the play-out	RTP/UDP	UDP is often blocked by firewalls	No waiting Time
			IP multicast		Low latency
			IP unicast		Real-Time delivery
Adaptive Streaming	Web-based TV	Download of small chunks or segments of media during play-out	Multiple HTTP-based solutions	Media content pre-processing (Chunks) for various quality formats	Reduced waiting time. Adaptation to the client's requirements and network conditions

TCP streaming, by using traditional *stateless*[5] Web servers. In this context, different vendors and standardization bodies have specified their own HAS solution, such as: HTTP Live Streaming (HLS) by Apple [31], HTTP Dynamic Streaming (HDS) by Adobe [32], Microsoft Smooth Streaming Solution (MS-SSTR) by Microsoft [33], and MPEG Dynamic Adaptive Streaming over HTTP (MPEG-DASH) by ISO/IEC and MPEG Group [4].

The main characteristics of managed and unmanaged streaming, a comparison between them, and their suitability in different scenarios can be found in [29] and [30].

Table III lists and compares the four main broadband delivery methods: downloading, progressive downloading, streaming and adaptive streaming. Downloading requires the full download of the media file prior the play-out. Thus, it has the highest waiting time. Its main advantages are the continuous play-out and the unnecessary buffering techniques. Streaming, particularly used in IPTV, is the technique used for real-time media delivery that guarantees a reduced waiting time and low latency with a reduced buffer size. Progressive downloading, mainly used in Web-based TV, is half way from downloading to streaming. It reduces the waiting time due to the download of the media file during its play-out, but it is not real-time delivery as streaming is. The final and most recent method, also used by latest Web-based TV and IPTV solutions, is Adaptive Streaming, which provides an adaptive play-out according to end-user's requirements and network conditions, by switching between representations of media streams (i.e., different versions of the media encoded with different quality formats). It can also be seen as something between streaming and downloading. It aims to achieve the benefits of both media delivery techniques by downloading independent and subsequent media chunks. The chunks are small media file segments in which media is fragmented, each one containing a short interval (between 2s to 10s long) of play-back time. The HTTP server provides the chunks of the same content at a variety of different bit rates comprising sequenced short play-back time intervals. During the play-out, the client dynamically selects (client-driven) the next file to download from the alternatives based on its current network conditions or requirements, minimising waiting time, achieving low latency, reducing the buffer's size and guaranteeing smoothness and continuity of media play-out, which are the main streaming benefits.

In this paper, we focus on MPEG-DASH (Section III-G), which is the solution proposed by MPEG and has also been adopted by HbbTV specification [34].

The transport protocol RTP is the traditional media delivery protocol for real-time media delivery providing timestamping and sequence number whereas RTCP, its companion, provides media delivery monitoring tools, minimal control and identification functionality [3]. More details about the protocol are found in Section V.

D. Standards

The accomplishment of (multiple streams) media sync requires an in-depth knowledge of how different video and audio MPEG standards convey timelines over IP Networks, as well as the protocols used for media delivery. The standards studied in this paper are MPEG-1, MPEG-2, MPEG-4, MPEG-DASH, ISO BMFF and the latest MMT standard. Moreover the RTP/RTCP is also included as a real-time media delivery transport protocol. The MPEG-2 part 1 is especially relevant because it is the main packetized system and media container used by most of the subsequent standards.

There are three main ISO/IEC MPEG standards: ISO/IEC 11172, 13818 and 14496. They are known as MPEG-1, MPEG-2 and MPEG-4, respectively. All of them are divided into parts, where specific areas are extended. MPEG-1 has 5 parts, MPEG-2 has 11 parts and, finally, MPEG-4 has 27 parts.

In all of them, Part 1 'systems' specifies the storage and transmission of multiple media streams along with the mechanism to facilitate synchronized decoding; Part 2 'video' explains the video coding method; Part 3 'audio' specifies the audio coding method; Part 4 'compliance/conformance testing' describes the test required to verify the proper bitstream production by encoders and the correct decoder's behaviour; and, finally, Part 5 'software simulation/reference software' establishes the software references to develop encoders and decoders.

The MPEG core sections are the audio and video encoding

[5]Server that do not retain any information about clients' state

systems and file formats. Table IV presents a summary of audio/video codecs and media file formats within MPEG Standards. A large selection of codecs for video and audio are published, although their study is outside the scope of this paper.

DVB, independently of the delivery platform being used, performs the media delivery via MPEG-2 Systems (DVB Transport Streams). Therefore, both of them work hand in hand. To achieve a correct decoding process, program and services information are encoded using MPEG-2 Program-Specific Information (MPEG-2 PSI) and DVB Service Information (DVB SI) tables.

MPEG-2 PSI and DVB SI are also used to deliver/provide time information within DVB streams. A thorough description of MPEG-2 PSI and DVB SI and the associated tables is detailed in Section IV.

The technical specification ETSI TS 102 823 [35] provides a means to synchronize DVB transport streams. This solution adds one or multiple broadcast timelines within the DVB stream via MPEG-2 Transport Stream (MP2T) packets. It applies the insertion of descriptors conveyed within the *auxiliary_data_structure*. In Section IV-A this solution is explained with further details for DVB Systems.

MPEG has proposed MPEG-DASH, which is further explained in Section III-G. A solution for Hybrid Digital Media Content sync using ETSI TS 102 823 [35] is presented in [36], using MPEG-DASH as a broadband adaptive streaming method. The proposed system implements a solution to generate and insert the broadcast timeline within the DVB MP2T stream.

HbbTV standard specifies the protocols used by the broadband or broadcast delivery platforms. Digital Storage Media Command and Control (DSM-CC) is used in broadcast, whereas broadband protocols include HTTP for unicast delivery and download, and MPEG-DASH for streaming.

MMT [9] is the latest approved MPEG media delivery standard for heterogeneous networks. It is intended to provide solutions for the latest challenges regarding media delivery, which consists of content access anywhere from a large number of devices via heterogeneous networks. In Section III-H MMT is further described.

TABLE IV: Video and Audio Codecs within MPEG Standards

Standard	Video	Audio	Media File Format
MPEG-1	MPEG-1 part 2	MPEG-1 Layer 1 (MP1)	MPEG-1 part 1
		MPEG-1 Layer 2 (MP2)	
		MPEG-1 Layer 3 (MP3)	
MPEG-2	H.262 part 2	MPEG-2 Layer 3 (MP3)	MP2T part 1
		AAC part 7	MP2P part 1
MPEG-4	H.263 part 2	HE-AAC part 3	ISO part 12
	H.264/AVC part 10		MP4 part 14
	Web Video Coding part 29		AVC part 15

III. Time and Timing within MPEG Standards

In this Section the description of the techniques used by MPEG Standards to synchronize encoder and decoder clocks, thus enabling synchronized play-out, is presented.

One of the most important concepts specified in MPEG-2 Systems is the Transport Stream (TS) concept, which is the media container used by MPEG-1, MPEG-2 and MPEG-4 to stream media over multiple distribution systems (with probable error occurrences).

To achieve intra- and inter-media sync, timestamps and clock references are used by all MPEG standards. In each standard, the timestamps and clocks references are stored in different fields located in different headers and each can have different resolution, frequency and constraints.

Other important concepts in all MPEG Standards are Elementary Stream (ES), Packetized ES (PES) and Program Stream (PS). An ES is a stream of one encoded media type (e.g., video or audio). These media streams are packetized in Packs in MPEG-1 or in PES in MPEG-2, further explained in Section III-C and III-D, respectively. These Packs/PES are associated with systems' information (such as time) to be transformed into PSs or TSs.

We firstly introduce the meaning and functionality of timestamps and clock references to further explain how they are included within each MPEG standard.

A. Clock References

Clock references within MPEG standards relate only to the encoder's relative media clock (no global clocks or time references are used). Such references are the means used by MPEG standards to reproduce encoder's clock rate at the decoder. In other words, they are the mechanism to recreate encoder's clock frequency at the decoder to guarantee the correct media stream play-out. If both clocks are running at the same frequency and have a common initial reference time, then timestamps will relate exactly to the same moment in time. As previously discussed, both time and timing affect media sync.

Clock references are needed because any timestamp used by the media source/s is based on the encoder's clock. For example, to accomplish the correct play-out of the audio and video streams, as well as the expected sync between audio and video (i.e., lip-sync), the audio and video decoder's clocks need to accurately reproduce the audio and video encoder's ones, respectively.

B. Timestamps

All MPEG standards related to audio/video deal with inter-media sync via timestamps. A timestamp field is used to agree on a specific moment in time, such as decoding or playing time. In the lip-sync example, timestamps within an stream are conveyed to synchronize the playing moment of an audio and a video AUs, so the video stream is displayed synchronized with the audio stream.

Different MPEG standards define different timestamps, but

TABLE V: Terms for MPEG-1 timelines related to Section III-C. Definitions from [38]

term	Meaning
i	'Index of any byte on the pack, including the pack header'
i'	'Index of the final byte of the SCR field in the pack header'
DTS	'Intended time of decoding in the STD of the first AU that commences in the packet'
j	'index to AU in the ESs'
k	'index to PU in the ESs'
PTS	'Intended time of presentation in the STD of the PU that corresponds to the first AU that commences in the packet'
SCF_{mpeg1}	'Frequency of a clock meeting these requirements'
SCR(i)	'Time encoded in the SCF field in units of the system clock'
$td_n(j)$	'The decoding time of AU $A_n(j)$'
$tp_n(k)$	'The presentation time of Presentation Unit (PU) $P_n(k)$'
tm(i)	'it is the time, measured in seconds, encoded in the system_clock_reference of pack p'

the general concept that applies to all of them is that timestamps refer to agreed moments in time for a specific purpose. In every MPEG standard two types of timestamps are defined. The first one is the Decoding Timestamp (DTS), which is common to all standards. The second one can be either the Presentation Timestamp (PTS), in MPEG-2, or the Composition Timestamp (CTS), in MPEG-4. These timestamps will be discussed later for each particular standard.

The need for two different timestamps is caused by the presence of different types of video frames, such as intra (I-frame[6]), Predicted (P-frame[7]) and Bi-predictive (B-frame[8]) frames. B-frames are encoded using the previous and the subsequent I/P-frames. Therefore, I/P frames may have to be decoded previous to their presentation time to be accessible for the B-frames decoding process. In other words, those I or P-frames will have DTS different from PTS/CTS to be decoded prior to their presentation or composition time, thus being available for any B-frame linked to them.

This can be appreciated in Fig. 3, which shows an example of a distribution of I, B and P-frames within a Group of Pictures (GOP) and the links between these frame types. For example, the DTS of P-frame$_4$ is previous to the DTS from B-frame$_2$ and B-frame$_3$, respectively (and previous to their own PTS). This is because these B-frames need the I/P frames they depend on to be previously decoded.

When a video stream only conveys I and P-frames, these frames would have DTS equal to PTS because, even in the case of a P-frame, any I-frame it depends on would be previously decoded at the receiver. In the case of audio, DTS always equals PTS due to the absence of different types of frames.

[6]'*Pictures that are coded using information present only in the picture itself and not depending on information from other pictures*' [37]. As example in Fig. 3 P_1 is coded without the reference of any other frame

[7]'*Pictures that are coded with respect to the nearest previous I or P-picture*' [37]. As example in Fig. 3 P_4 is coded with the reference of I_1

[8]'*Pictures that use both future and past pictures as a reference*' [37]. As example in Fig. 3 B_2 is coded with the reference of I_1 and P_4

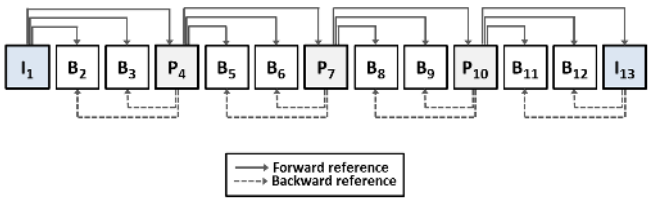


Fig. 3: GOP containing I, P and B-frames

C. MPEG-1

When MPEG-1 was standardised, it was only intended to be a storage medium for video and audio data. Thus, transport over IP networks was not considered. Later on, when MPEG-2 presented a solution to transport media streams over IP Networks, it also included a solution for MPEG-1, as we further explain in Section III-D.

MPEG-1 Program Streams (PS) are stored in packs. An ES is packetized in a variable number of packets which are conveyed into packs (See Fig. 4). Each pack contains certain fields with timing information, a system header and multiple packets where ES data are conveyed. The structure of MPEG-1 PS with all the time related fields is outlined in Fig. 5.

The System Clock Reference (SCR) field, which is included in the first pack (Pack1 in Fig. 5) of a sequence (encoded in its Pack Header, as can be seen in Fig. 5), will be used to set the decoder's clock to the encoder's. Moreover, due to clock drift, SCR values should be sent at a maximum time interval of 0.7s to allow the decoder to re-sync to the encoder.

According to [38], the SCR field '*indicates the intended time of arrival of the last byte of the system_clock_reference fields at the input of the System Target Decoder (STD)*'.

Consider that byte *i'* of the multiplexed stream enters the STD at time $t_m(i')$. Then, the time can be recovered by decoding the SCR fields, within the pack header, at the decoder's input stream.

The SCR(i') encoded value represents the time $t_m(i')$, where *i'* relates to the last byte of the SCR field.

$$SCR(i') = NINT(SCF_{MPEG1} \cdot (t_m(i'))) \% 2^{33} \quad (1)$$

SCR(i') is the time encoded in the 33-bit SCR[9] field measured in units of the System Clock Frequency (SCF). SCF_{MPEG1} is 90KHz. NINT stands for the Nearest Integer Function.

To reconstruct the time when any byte *i* within the multiplexed stream arrives at STD, input arrival time (*tm*), eq. (2) is applied [38].

$$tm(i) = \frac{SCR(i')}{SCF_{MPEG1}} + \frac{i - i'}{mux\_rate * 50} \quad (2)$$

In the previous equation *i'* represents the index of the final byte of the SCR fields in the pack header, and *mux_rate* represents the rate at which data arrives. The terms used in the equations in this sub-section are defined in Table V.

The 22-bit *mux_rate* specifies the rate at which a multiplexed stream enters the STD during the pack in which it

[9]SCR in MPEG-2 Program Stream (MP2P) is 42-bit value from 33-bit *SCR_base* and 9-bit *SCR_ext* fields at 27MHz frequency

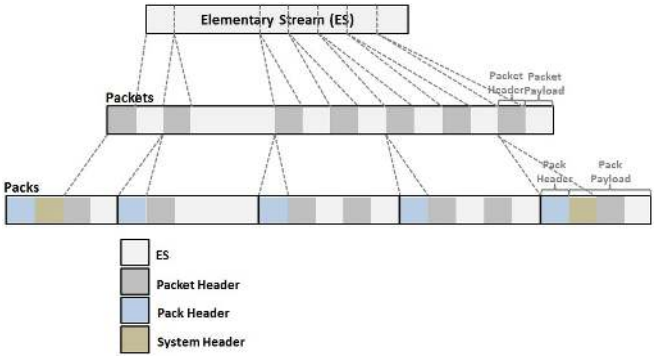


Fig. 4: ES Packetization process into MPEG-1 PS Stream (Packs)

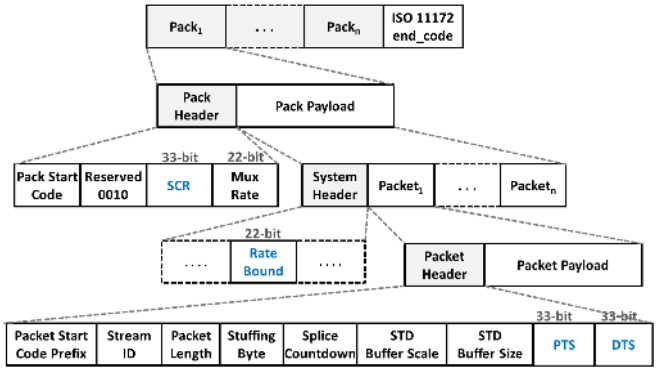


Fig. 5: MPEG-1 PS bitstream and its time related fields

is included. The unit of measurement is 50 bytes/s, rounded upwards. This field provides variable rate operation of the STD (its values can change pack to pack). The 22-bit *rate_bound* field (see Fig. 5) indicates the maximum value of the *mux_rate*. The decoder uses *rate_bound* to verify its capability to decode the stream.

In this case, PTS and DTS timestamps values are expressed with 33-bit resolution at a 90KHz frequency[10]. The PTS '*indicates the intended time of presentation in the STD of the presentation unit that corresponds to the first AU that commences in the packet*' while the DTS '*indicates the intended time of decoding in the STD of the first AU that commences in the packet*' [38]. A video AU begins if the '*first byte of a video picture start code*[11] *is present*' [38] in the data packet. Similarly, an audio AU commences if the first byte of the sync word of an audio frame is present in the data packet [38].

A PTS is only present in the packet header if the payload carries an ES containing the first byte of a picture start code, for video, or the first byte of an audio AU, for audio. A DTS is present in a packet header given the two following requirements: a PTS is also present in the packet header, and the decoding and presentation time are not equal.

DTS and PTS can be calculated from the SCF, rate at which SCR increments, using the following equations[12] from [38]:

$$PTS = NINT(SCF_{MPEG1} \cdot (tp_n(k)))\%2^{33} \quad (3)$$

$$DTS = NINT\left(SCF_{MPEG1} \cdot (td_n(j))\right)\%2^{33} \quad (4)$$

In eq. (3), parameter $tp_n(k)$ is the presentation time (in seconds) in the STD of the k^{th} presentation unit, which is the one associated to the first AU (of the n_{th} elementary stream ES_n) that commences in the packet data. In eq. (4) parameter $td_n(j)$ is the decoding time (in seconds) in the STD of the first AU that commences in the packet data which is the j^{th} AU (of the ES_n) [38].

SCR is conveyed in every single pack whereas timestamps, PTS and DTS, are not. However, consecutive decoding times of AUs without encoded DTS or PTS fields can be obtained from information in the ES. SCF_{MPEG1} is 90KHz, but the following constraints are established in [38]:

$$90kHz - 4.5Hz \leqslant SCF_{MPEG1} \leqslant 90kHz + 4.5Hz \quad (5)$$

This expression provides the maximum and minimum possible values of SCF_{MPEG1}. Changes can be applied to correct the SCF_{MPEG1} to ensure it is always within the boundaries. Nevertheless, the rate of changes should not be greater than $250 \cdot 10^{-6}$ Hz/s [38].

$$SCF_{MPEG1}ChangeRate \leqslant 250 \cdot 10^{-6}Hz/s \quad (6)$$

D. MPEG-2

MP2T are used to transport MPEG-1, MPEG-4 and Advanced Video Coding (AVC) streams. First, in the specification of MPEG-2 part 1 (in 1996), transport of MPEG-1 streams was included. Second, after the MPEG-4 approval, an addition was made to transport MPEG-4 and AVC [21].

MP2T and MPEG-2 Program Stream (MP2P) are fully specified in MPEG-2 part 1, 'Systems'. MP2P is related to the previous described MPEG-1 PS, both used for storage purposes [21].

The MPEG-2 'Systems' part describes the STD, which implements sync and buffering methods related to the media streams. Sync takes place at the decoding and presentation stages, while buffering techniques need to ensure that neither buffer overflow nor underflow occur.

PSs are described in MPEG-1 and MPEG-2, whereas TSs are only described in MPEG-2 Systems. MP2P is designed for error free applications, such as storage, and MP2T for transport over multiple distribution systems (with possible error occurrences). Unlike in MPEG-1, in which packets have variable size, a MP2T multiplex is made up by fixed 188-byte length packets called '*transport packets*' (TS packets). Each transport packet contains a 4-byte header, an optional Adaptation Field (AF) and the Payload (the MP2T structure is found in Fig. 6a, whereas the PES structure is shown in Fig. 6b).

This paper focuses on MP2Ts, which are used for media streaming over IP Networks. ESs (e.g., audio or video streams)

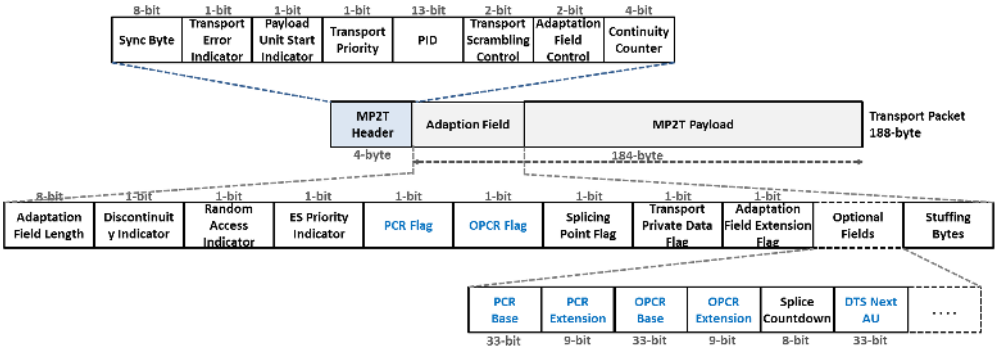
[10] 24hours/day * 60min/hr * 60sec/min * 90k/sec (clock)=7776000k which needs 33 bits to be represented

[11] '*Start codes are specific bit patterns that do not otherwise occur in the video stream*' [39]. Multiple type of start codes are defined in [39]

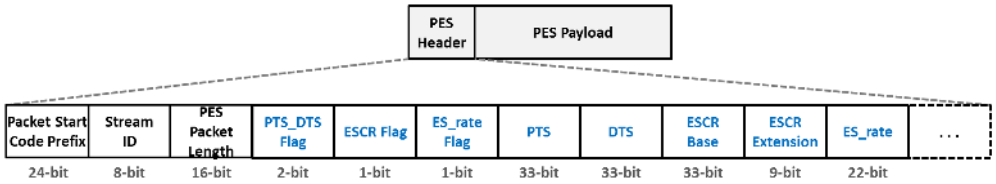
[12] DTS and PTS equations in MPEG-1 do not use sub-indexes, which differs from MPEG-2 PS

Fig. 6: MP2T and PES packet structure

(a) MP2T packet structure



(b) PES packet structure



are packetized into PES and each PES is divided in TS packets. An MP2T is generated by multiplexing TS packets from the PES of one or several programs. Fig. 7 shows the process from an ES to a MP2T stream (with only one ES).

On the one hand, MP2P constitutes one PS with a unique time base, and with SCR as its clock. On the other hand, MP2T conveys multiple PS, each with a different time base, and, therefore, each PS having its own independent PCR. As a result, clock references have different constraints. They shall be encoded at least every 0.7s for MP2P and at least every 0.1s for MP2T (meaning the coding frequency) [21].

A PES is transported within multiple MP2T transport packets. As can be seen in Fig. 6a, each transport packet can have an AF before the PES data (transport packet payload) and just after the MP2T header. Both PES and AF convey timing information. The latter carries stream information, while the former conveys the media data and information within the PES header.

An MP2T stream entering the STD contains several programs, each of them with a independent time base. However, only one program within the MP2T is decoded at a time. The MP2T stream enters the STD at a constant rate [21]. The PCR field defines '*the time t(i) at which the i^{th} byte enters the T-STD*' [21], taking into account the number of bytes between consecutive PCRs fields. In Table VI the terms used in all the equations in this sub-section are defined.

In Fig. 6a, we can see the 188-byte size MP2T transport packet format. The AF time related fields, PCR and Original Program Clock Reference (OPCR), are shown at the bottom of the figure. The 1-bit *OPCR_flag* field signals the presence of the OPCR field, in the same way as the 1-bit *PCR_flag* field indicates the presence of PCR field.

As shown in Fig. 6a, the clock reference is conveyed in the AF, divided into two fields: the 33-bit *program_clock_reference_base* (PCR_base) field in units of the period 1/300 times the SCF, and the 9-bit *program_clock_reference_ext* (PCR_ext) field in units of the SCF. Both fields are part of the PCR, clock reference which always runs at 27MHz SCF in MP2T, named SCF_{MPEG2}. The constraints of SCF_{MPEG2} are the following:

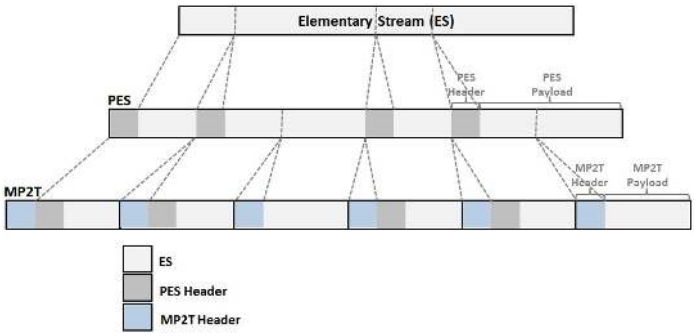


Fig. 7: ES Packetization process into MP2T stream

$$27MHz - 810Hz \leqslant SCF_{MPEG2} \leqslant 27MHz + 810Hz \quad (7)$$

$$SCF_{MPEG2}ChangeRate \leqslant 75 \cdot 10^{-3} Hz/s \quad (8)$$

The encoded value in the PCR field represents the time *t(i)*, when the byte within the MP2T ('*containing the last bit of the program_clock_reference_base fields*' [21]) arrives at the STD. The following equations are applied [21].

$$PCR(i) = PCR_{base}(i) \cdot 300 + PCR_{ext}(i) \quad (9)$$

where

$$PCR_{base}(i) = \left(\frac{SCF_{MPEG2} \cdot t(i)}{300} \right) \%2^{33} \quad (10)$$

TABLE VI: Terms for MPEG-2 timelines related to Section III-D. Definitions from [21]

term	Meaning
CA_{freq}	Clock Accuracy Frequency
CA_{ext}	'Together with the CA_integer, it gives the fractional frequency accuracy of the system clock in parts per million'
CA_{int}	'Together with the CA_exponent, it gives the fractional frequency accuracy of the system clock in parts per million'
DTS(j)	'it indicates the decoding time, $td_n(j)$, in the STD of an AU *j* of ES_n'
i	'index of any byte in the Transport Stream for i"<i<i' '
i'	'index of the byte containing the last bit of the immediately following PCR_base field applicable to the program being decoded'
i"	'index of the byte containing the last bit of the most recent PCR_base field applicable to the program being decoded'
j	'index to AU in the ES'
k	'index to PU in the ES'
n	'index to the ESs'
PCR(i)	'it indicates the time t(i), where *i* is the index of the byte containing the last bit of the PCR_base field'
PCR_{base}	'in units of the period of 1/300 times the system clock frequency'
PCR_{ext}	'units of the system clock frequency'
PTS(k)	'indicates the time of presentation, $tp_n(k)$, in the STD of a PU *k* of ES *n*'
SCF_{MPEG2}	System Clock Frequency of a MPEG2 program
$td_n(j)$	'decoding time of AU $A_n(j)$'
$tp_n(k)$	'presentation time of PU $P_n(k)$'
TR(i)	'number of bytes in the Transport Stream between the bytes containing the last bit of two successive PCR_base fields of the same program divided by the difference between the time values encoded in these same two PCR fields'

$$PCR_{ext}(i) = \left(\frac{SCF_{MPEG2} \cdot t(i)}{1} \right) \%300 \qquad (11)$$

Considering *i*, *i'*, *i"* as indices to bytes in the MP2T (i"<i<i', and the first byte of the MP2T having index 0), eq. (12) is applied to find the time when any byte *i* within the TS arrives at the STD (input arrival time) [21]:

$$t\,(i) = \frac{PCR\left(i''\right)}{SCF_{MPEG2}} + \frac{i - i''}{TR\,(i)} \qquad (12)$$

where parameter i" is the index of the byte containing the last bit of the latest *PCR_base* field. PCR(i") is the encoded time in the *PCR_base* and *PCR_ext* fields in system clock units. The Transport Rate (TR), *TR(i)* is the TR for any byte *i* between bytes *i"* and *i'* can be derived from PCR values and the SCF (27MHz in MP2T), as shown in eq. (13) [21].

$$TR\,(i) = \frac{((i' - i'') \cdot SCF_{MPEG2})}{PCR\,(i') - PCR\,(i'')} \qquad (13)$$

where '*i' is the index of the byte containing the last bit of the next PCR_base fields*' [21], related to the program being decoded. TR in [21] is defined as '*the number of bytes in the transport stream between the bytes containing the last bit of two successive program_clock_reference_base fields of the same program divided by the difference between the time values encoded in these same two PCR fields*'.

To reconstruct an original single program from an MP2T, a 42-bit OPCR field is used. In this process, OPCR is only present in the MP2T packets in which PCR is found. OPCR is a replica of its mapped PCR in the original MP2T program, following the same exact structure (fields, resolution, and number of bits) as PCR. OPCR, as PCR, consists of two fields: the 33-bit *original_program_clock_reference_base* (OPCR_base) field and the 9-bit *original_program_clock_reference_extension* (OPCR_ext) field.

When PESs are not conveyed within MP2T, e.g., when PESs are directly conveyed within an RTP packet, the MP2T header and MP2T AF are not present and other clock reference needs to be conveyed within the PES packet. Such reference is the *elementary_stream_clock_reference* (ESCR), which is conveyed in the PES Packet header, as shown in Fig. 6b. Its presence is signalled by the 1-bit *ESCR_flag* field. ESCR follows the SCR and PCR characteristics, with 42-bit size (resolution), divided into 33-bit *ESCR_base* and 9-bit *ESCR_ext* fields. The ESCR field indicates the expected arrival time, at the PES stream associated STD, of the byte containing the last bit of the *ESCR_base* field.

The 22-bit *ES_rate* field specifies the rate at which the STD receives the PES stream bytes. The *ES_rate* validity range goes from the first packet including the field until the packet containing the following *ES_rate* value. Therefore, *ES_rate* values may differ in different PES packets. The units of the *ES_rate* measurement are 50bytes/second (being value '0' forbidden). PES timing information is encoded within the ESCR and *ES_rate* fields. ESCR will be '*used in place of the SCR and ES_rate in place of program_mux_rate*' [21].

Timestamps in MPEG-2 part 1 are included in DTS and PTS fields, both of them 33-bit size (resolution) and 90KHz frequency. The DTS and PTS are conveyed within the PES Packet Header and the 2-bit *PTS_DTS_flag* field indicates their presence. DTS and PTS are equally present in the MP2P and MP2T and have the same meaning as in MPEG-1.

As in MPEG-1, in MPEG-2 the equations to obtain the PTS and DTS timestamps are also based on the decoding and presentation times, in seconds, and the SCF_{MPEG2}. The values of both PTS and DTS are defined in units of the period of the SCF divided by 300 (compliant with 90kHz):

$$PTS(k) = \frac{(SCF_{MPEG2} \cdot (tp_n(k)))}{300} \%2^{33} \qquad (14)$$

$$DTS(j) = \frac{(SCF_{MPEG2} \cdot (td_n(j)))}{300} \%2^{33} \qquad (15)$$

PTS(k) indicates the time of presentation, in the STD, of the k^{th} presentation unit of ES_n. *DTS(j)* indicates the decoding time, in the STD, of the j^{th} AU of ES_n. Parameter $tp_n(k)$ in eq. (14) '*is the presentation time, measured in seconds, in the STD, of the k^{th} presentation unit in ES_n*' [21]. Parameters $td_n(j)$ in eq. (15) '*is the decoding time, measured in seconds, in the STD, of the j^{th} AU in ES_n*' [21].

One AU could be conveyed in multiple PES (common for video AUs) and multiple AUs can be carried in one PES

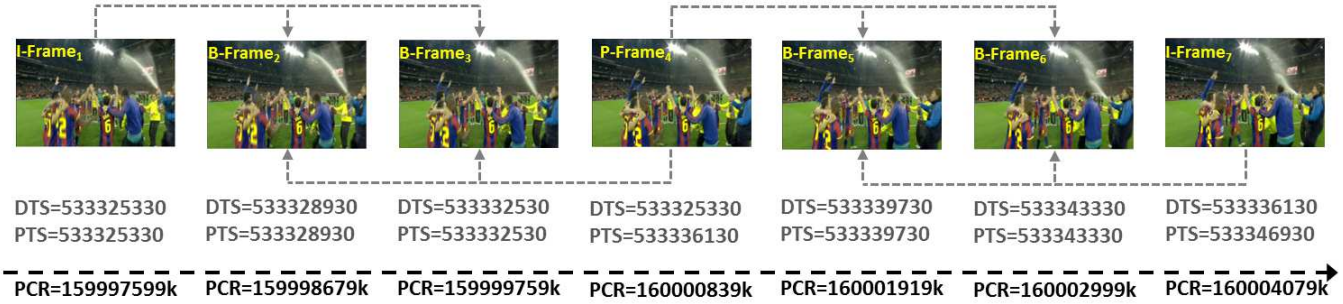


Fig. 8: Example of frame sequence with timestamps and PCR timeline values

TABLE VII: System Clock Descriptor Fields and Description [21]

Field	Bits	Description/Utility
Descriptor_tag	8	Value 11 for MP2P and MP2T. It signals the format of the System Clock Descriptor (SCD)
Descriptor_length	8	Descriptor bytes size after the *descriptor_length* field. It is useful to know the end of the descriptor
External_clock reference indicator_flag	1	Flag that indicates the use of a reference external clock. It indicates that an external clock was used to generate the timestamps
Reserved	1	-
Clock_accuracy integer	6	Integer of frequency accuracy of system clock, parts per million (ppm) units. It is used to calculate clock accuracy if it is higher than 30ppm
Clock_accuracy exponent	3	Exponent of frequency accuracy of system clock (ppm). It is used to calculate clock accuracy if it is higher than 30ppm
Reserved	5	-

(frequent for audio AUs). In both cases timestamps refer to the first AU within the PES packet.

An extra timestamp is also used in MP2T, the 33-bit *DTS_next_AU* field (90KHz frequency), which is used to support media streams splicing. Splicing is used to concatenate the end of a media stream with the beginning of another one. In the case of seamless splicing, the fields *splice_type* and *DTS_next_AU* are present. *DTS_next_AU* field denotes the decoding time of the first AU found just after the splicing point, and is located in the AF, whereas the 1-bit *seamless_splice_flag* field indicates its presence. *DTS_next_AU* field is only used in MP2T, but not in MP2P [21].

The only requirement for timestamp coding frequency is that the time interval between packets conveying PTS shall be less than 0.7s. DTS and *DTS_next_AU* have no requirements.

In Fig. 8 an example with DTS and PTS timestamp values of I, P, and B-frames is presented. The frames have a time interval between frames is 40ms (equivalent to 1080k PCR frequency units or to 3600 in timestamps frequency units). Seven consecutive video frames have been used to describe the timestamp process. The frame types in Fig. 8 are used as an example for timestamp purposes.

There are two other key tools that play an important role for media sync: the System Clock Descriptor (SCD), to provide extra clock information, and the Phase Lock-Loop (PLL), to reproduce encoder's clock frequency at decoder.

1) System Clock Descriptor (SCD): Descriptors are (generally) optional, variable-length data elements that can add standard-defined or user-defined data elements to MPEG-2 private table sections. SCD is utilized to transfer the encoder's system clock information, used in the timestamping process, to the decoder. It consists of several fields related to the clock accuracy, which are depicted in Table VII. It is conveyed within a MP2T packet as a descriptor of the Program Map Table (PMT)[13].

Clock accuracy fields are needed if more than 30ppm accuracy is required. The Clock Accuracy Frequency (CA_{freq}) is given by eq. (16) [21]:

$$CA_{\text{freq}} = \begin{cases} 30ppm & \text{if } CA_{\text{int}}=0 \\ CA_{\text{int}} \cdot 10^{-CAexp} & \text{if } CA_{\text{int}}\neq 0 \end{cases} \quad (16)$$

where parameter CA_{int} is the value of the 6-bit Clock Accuracy Integer field, and parameter CA_{Exp} is the value of the 3-bit Clock Accuracy Exponent field.

2) Phase-Locked Loop (PLL): PLL is the tool used by the STD to synchronize encoder's and decoder's frequency. Its main elements are shown in Fig. 9.

The PCR/SCR from the stream enters the PLL where it is compared with the decoder's System Time Clock (STC) by the substractor. The difference is then sent to the Low-Pass Filter and Gain, where the output frequency is calculated. Finally, the Voltage Controlled Oscillator (VCO) establishes the new SCF. Based on the new SCF, the STC Counter sets a new STC. System is locked, meaning process will be repeated, until the SCF is 27MHz.

E. MPEG-4

An MPEG-4 overview is given in [40], including its architecture, multiplexing and sync aspects. MPEG-4 is a layered model divided into three layers (Fig. 10): Compression, Sync and Delivery layers. The Compression Layer (CL) and the Sync Layer (SL), where time and timing information is conveyed, are independent from the Delivery Layer (DL), although the CL depends on the media type. Finally, the DL is media independent (a deep description of DL is provided in [41]).

The ES management is MPEG-4 is described in [42] and

[13]In Section IV MPEG-2 PSI tables are described.

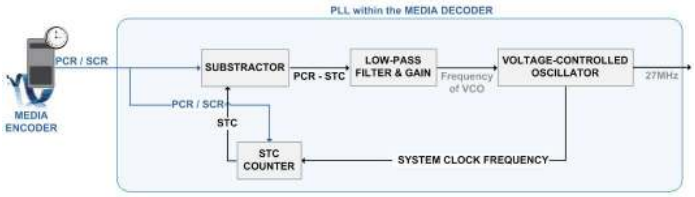


Fig. 9: MPEG-2 PLL [21]

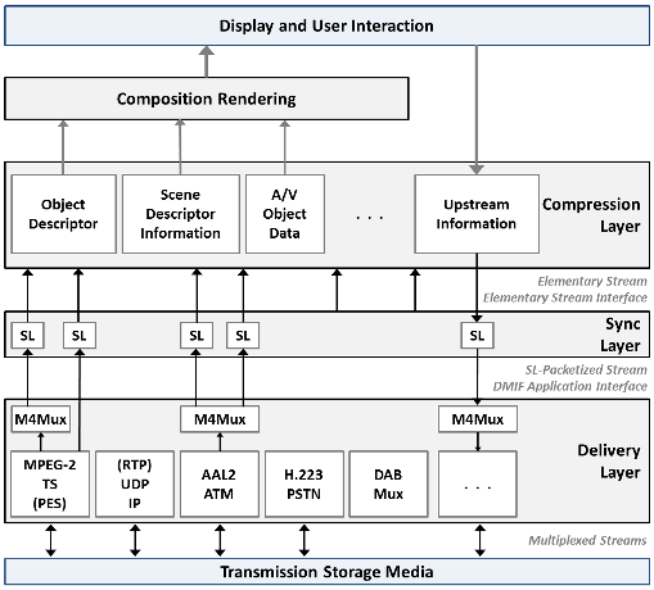


Fig. 10: MPEG-4 High Level Layers diagram [44]

[43]. It is important to define a shared mechanism to convey timing and framing information. SL is the sole mechanism defined for this purpose in MPEG-4. It is a packet-based interface (SL packet is the smallest data unit), i.e., a flexible and configurable packetization facility, which provides the tools to share information (including timing information) between the CL and DL layers [40]. SL provides the insertion of timing and framing information into the related data packets, i.e., complete AUs.

Any timing system shall be supported by MPEG-4 (low and high bitrates), thus the SL should be adjustable to accommodate all operational methods.

SL supports the configuration of size and resolution of timestamps and clock references to operate at all bitrates although the traditional clock recovery techniques using clock references and timestamps can also be used. A rate-based approach can be used rather than using explicit timestamps, as the known rate of the AUs implicitly determines their timestamps. A typical example of this is a slide-show presentation. However, the main operation mode incorporates the clock references and timestamps. The system decoder model facilitates the sync between the receiver and sender clocks and buffer resources management [40].

MPEG-4 is object oriented, therefore it is based on the definition of media objects representing a semantically meaningful audio or visual entities (timed and non-timed media data). Media objects are grouped into scene objects. MPEG-4

TABLE VIII: Terms for MPEG-4 timelines related to Section III-E. Definitions from [44]

term	Meaning
$AU_{duration}$	'the duration of an access unit' in timescale units
AU_{time}	AUduration in time units (seconds)
CTS	'Each CU has an associated nominal composition time, the time at which it must be available in the composition memory for composition'
CU_{time}	CUduration in time units (seconds)
$CU_{duration}$	'the duration of a composition unit' in timescale units
DTS	'Each AU has an associated nominal decoding time, the time at which it must be available in the decoding buffer for decoding.'
FCR(i")	'is the time encoded in the *fmxClockReference* in units of FCRResolution'
FCR_{res}	is the resolution of the *fmxClockReference* in cycles per second
fmxRate(i)	'indicates the rate specified by the fmxRate field for byte *i*'
i	'is the index of any byte in the M4Mux stream for i"<i<i' '
i"	'is the index of the byte containing the last bit of the most recent *fmxClockReference* field in the M4Mux stream'
k	'k is the number of times that the objectClockReference counter has wrapped around'
m	'an integer value denoting the number of wrap-arounds' for timestamps values
$SL.OCR_{len}$	'is the length of the objectClockReference field in SL packet headers'
$SL.OCR_{res}$	'is the resolution of the object time base in cycles per second'
SL.timescale	'used to express the duration of access units and composition units. One second is evenly divided in timeScale parts'
$SL.TS_{len}$	'is the length of the time stamp fields in SL packet headers'
$SL.TS_{res}$	'is the resolution of the time stamps in clock ticks per second'
$t_{estimated}$	'current estimated value of the OTB'
$t_{OTBrec}(k)$	OTB reconstructed time for value *k*
$t_{ts}(m)$	Timestamp for value *m*

systems specify the relations between a scene object and all the media objects that compose the scene. Media objects are carried into one or more ES [43]. In Table VIII the terms used in the equations used in this sub-section are defined.

Fig. 11 shows an example of a scene, in which a frame with two Video Objects (VOs) can be appreciated: one football player and the background (the grass). The AUs are waiting in the Decoding Buffers (DB_1 and DB_2). VOs are decoded at DTS time, td_1 (football player) and td_2 (background), and, once objects have been decoded, the Composition Units (CUs) wait in the composition buffer (CB_1 and CB_2) until their composition time (tc_1 and tc_2).

The entire frame/picture is considered a CU. In MPEG-4 the description of the scenes is organized in two levels: the structural level and the media object description level. The structural level includes the Binary Format for Scene (BIFS) which specifies how the media objects are organised in time and space within a scene object. On a lower level, the

media object description framework specifies the location of different media streams, their configuration and how they are synchronized [43].

The scene description and its associated Object Descriptors (OD) are essential to access an MPEG-4 presentation. Object and Scene descriptors are carried in individual ESs separately from the MPEG-4 presentation. Other important auxiliary information is also carried in other ESs, such as the Clock Reference Stream (described in Section III-E3). Object Descriptors are encapsulated in messages using a lightweight protocol [43]. These data are comparable to MPEG-2 PSI or DVB SI in MPEG-2 applications.

In this section, we only focus on the time and timing model in the SL packetization process, and on the M4Mux tool, which is a low overhead and low delay tool designed for interleaving SL streams with flexible instant bitrate.

1) SL Packetization: Time and timing in MPEG-4 are conveyed via timestamps and clock references, in the same way as in MPEG-1 and MPEG-2, although MPEG-4 part 1 aims to be independent of the DL. To accomplish this independence, MPEG-4 adds the SL with the purpose of synchronizing the AUs and the CUs at the STD.

In MPEG-4, Composition Timestamps (CTS) are used (instead of PTS used in MPEG-2). An ES is a sequence of AUs containing DTS and CTS timestamps. The CTS indicates the composition time (presentation time in MPEG-2) when the different AUs should be composed and presented. An AU is decoded at DTS time, generating a CU which is presented at CTS time (see Fig. 11).

Time dependences between ESs are defined to allow the sync of several streams (inter-media sync). For example, the Scalable Video Coding (SVC) in MPEG-4 consists of a base layer and multiple enhancement layers. All layers related to the same media object share the same time base. Furthermore, different media objects share the Object Time Base (OTB) to perform inter-object sync (i.e., inter-media sync).

The SL Layer defines the format of the SL packets and the *SL Config Descriptor*. In the former, SL packet header conveys the time information (clock references and timestamps) about the media stream within the SL packet payload. The latter is a part of the ES descriptor that exists for each ES and is used to deliver configuration information about the SL stream fields, such as the length and resolution of the time related fields. The clock references, explained later in this section, can be derived from information in different fields, all encoded at the *SL Config Descriptor*.

Timing is delivered using clock references, which signal the encoder's clock frequency. Some applications may require that multiple encoder's share the same clock. Thus, it is possible to relate to clock references from another ES as well, given that MPEG-4 provides the means to create a special ES, with no media payload, that only conveys timing information, called *Clock Reference Stream* (further explained in Section III-E3) [43].

In MPEG-4, OTB at the encoder is transmitted via the Object Clock Reference (OCR) to synchronize the decoder with the receiver's System Time Base (STB). OCR is the clock reference for MPEG-4 (see Fig. 12). The frequency and

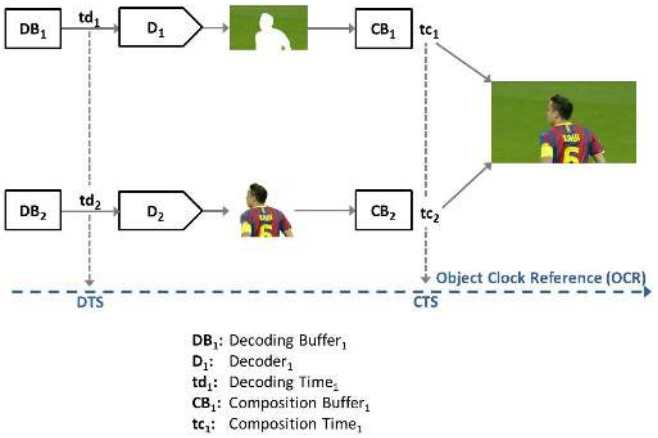


Fig. 11: Example of the Object High Level concept for MPEG-4 clock references (OCR) and timestamps (DTS/CTS)

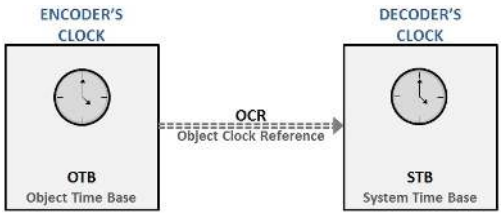


Fig. 12: MPEG-4 Clock References Location

number of bits of the OCR is flexible and are encoded at the *SL Config Descriptor* within the 32-bit *OCRresolution* (OCRres) and 8-bit *OCRlength* (OCRlen) fields. OCR is only present in the SL packet header if *OCR_flag* is set.

The OTB time value t_{OTB} is reconstructed from the OCR timestamp, according to the following equation [44]:

$$t_{OTBrec} = \left(\frac{OCR}{SL.OCR_{Res}} + k \cdot \frac{2^{SL.OCR_{Len}}}{SL.OCR_{Res}} \right) \quad (17)$$

where k is the number of times the OCR counter has wrapped around (number of times the value reaches the maximum and starts over). SL prefix indicates values conveyed within the SL Config Descriptor.

Every time an OCR is received, some steps shall be taken to also prevent k ambiguity. When the first OCR for an ES is acquired, the value of k shall be set to 1. For every subsequent OCR received, the current estimated value of OTB shall be sampled ($t_{OTBestimated}$), and then the value of t_{OTB} for different values of k shall be evaluated ($t_{OTBrec}(k)$). The value of k that minimizes the expression:

$$|t_{OTBestimated} - t_{OTBrec}(k)| \quad (18)$$

will be obtained and used to reconstruct t_{OTBrec} by using Eq. (17) [44].

Timestamps encoded in the SL packet header are used to synchronize the functions executed by the STD. The DTS encodes the instant in time when an AU shall be decoded, whereas the CTS encodes the instant in time when a CU shall be composed. Different AUs from one or multiple streams may be needed to compose a single CU. Both timestamps are

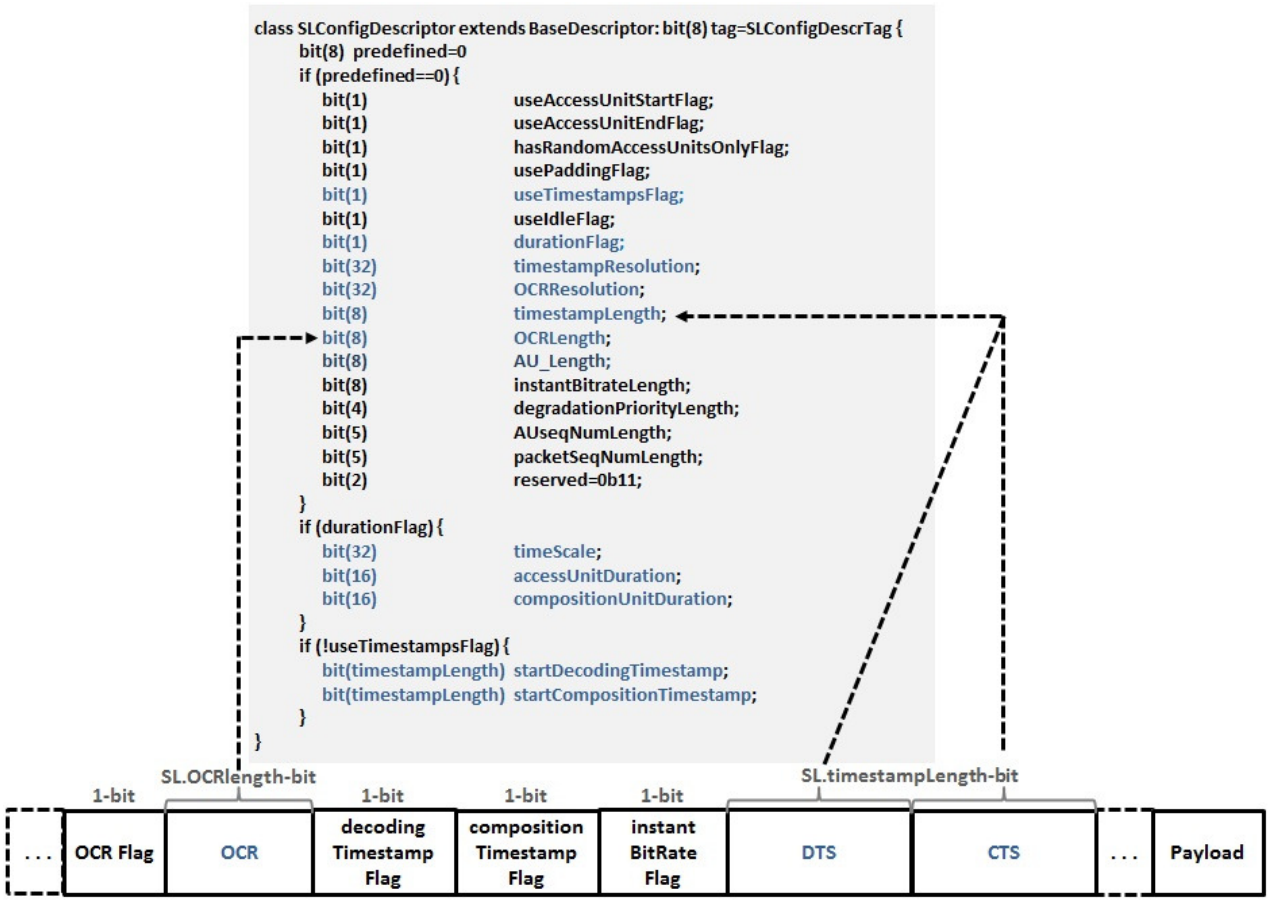


Fig. 13: MPEG-4 Part 1 bitstream and its time related fields and descriptors

carried in the SL packet header, although their size and resolution is indicated within the SL Config Descriptor in the 32-bit *timestampResolution* (TSres) and 8-bit *timestampsLength* (TSlen) fields, which apply to both types of timestamps, DTS and CTS. In Fig. 13, the SL stream structure and related *SL Config Descriptor* are depicted, focusing on the time related fields and information.

In previous Fig. 11, an example of the principles of DTS, CTS and VO is drawn. In the figure, the objects are displayed after being decoded (left part) at DTS time instant. Then, at CTS instant all objects compose the complete frame. Both timestamps instants, DTS and CTS, are related to the OCR clock reference timeline showed at the right part of the picture.

Occasionally, AUs contain a constant duration value of media data in time units. In such a case, timestamps can be obtained using different fields defined in the *SL Config Descriptor* (Fig. 13). In particular, AU duration (AUduration) and CU duration (CUduration) fields are used when the AU and CU contain a constant value of media data in time units (constant time duration).

The 32-bit *timescale* field is used to calculate the CU and AU time duration in seconds. The values included in the 16-bit *accessUnitDuration* (AUduration) field and in the 16-bit *compositionUnitDuration* (CUduration) field are divided by the value of *timescale* to calculate the AU and CU time in seconds, as can be seen in eq. (19) and (20), respectively:

$$AU_{time} = SL.AUduration \cdot \left(\frac{1}{SL.timescale} \right) \quad (19)$$

$$CU_{time} = SL.CUduration \cdot \left(\frac{1}{SL.timescale} \right) \quad (20)$$

In this case, two additional fields included in the SL Config Descriptor, *startDecodingTime* and *startCompositionTime*, containing the decoding/composition time of the first AU/CU within the ES, which are used to calculate the decoding and composition timestamps. The resolution of both fields corresponds to *TSres*.

The timestamp values are calculated by using the length (given by *TSlen* field) and resolution (given by *TSres* field) of the timestamps conveyed within the *SL Config Descriptor* [44]. The decoding time (t_D) of an AU is reconstructed from the DTS according the equation:

$$t_D = \left(\frac{DTS}{SL.TS_{Res}} + m \cdot \frac{2^{SL.TS_{Len}}}{SL.TS_{Res}} \right) \quad (21)$$

while the composition time (t_C) of the first CU resulting from that AU is reconstructed from CTS according to the equation:

$$t_C = \left(\frac{CTS}{SL.TS_{Res}} + m \cdot \frac{2^{SL.TS_{Len}}}{SL.TS_{Res}} \right) \quad (22)$$

where *m* is the number of wrap-arounds of the DTS or CTS timestamps counters in both eq. (21) and (22).

Both timestamps, DTS and CTS, have limited length, thus their time values calculated using previous equations may become ambiguous.

Accordantly, every time a timestamp is received, some steps shall be taken to prevent *m* ambiguity. For every timestamp received, the current estimated value of the OTB ($t_{OTBestimated}$) shall be sampled and the timestamp will be evaluated for different values of *m*.

$$t_{ts}(m) = \frac{timestamp}{SL.TS_{Res}} + m \cdot \frac{2^{SL.TS_{Len}}}{SL.TS_{Res}} \tag{23}$$

The value of *m* that minimizes the following expression shall be assumed to yield the correct value to reconstruct the t_{ts} value which will be used to estimate the timestamps received (t_{ts} can be either t_D or t_C) [44]:

$$|t_{OTBestimated} - t_{ts}(m)| \tag{24}$$

2) Usage of Object Clock References (OCR) and Timestamps: According to [43], the OCR time resolution shall allow differentiating from two OTB moments in time with a greater difference than the value:

$$\frac{1}{SL.OCR_{res}} \tag{25}$$

OCR resolution (*OCRres*) should be high enough for the media player to synchronize more than one ES. On the other hand, timestamps resolution shall be high enough to synchronize AU/CU within a stream [44].

TSres greater than *OCRres* does not provide better discernment between timestamps. Moreover, if *OCRres* is greater than *TSres*, the STD system does not benefit of the full *OCRres*.

OCR bit length, established in *OCRlen* within the *SL Config Descriptor*, should be long enough to assure *k* unambiguous positioning of time events from a set of ES.

When, at a media player, the value of *k* is known, the OTB time is unequivocal. When the *k* factor cannot be obtained, the timestamps are ambiguous. This can cause malfunction of the buffer model and errors at the decoder.

3) Clock Reference Stream: To share the timing information between multiple streams a specific *Clock Reference Stream* can be used, declared by means of the object descriptor. A *Clock Reference Stream* is a dedicated stream with the only purpose of conveying OCR clock references. The SL Packet Header within a *Clock Reference Stream* is configured to only convey the OCR values. Therefore, only *OCRres* and *OCRlen* are present in the SL packet header.

As any other streams, the *Clock Reference Stream* also uses SL packetized streams but, it uses a specific configuration of parameters in the SL packet, by means of two additional descriptors: *Decoder Config Descriptor* and *SL Config Descriptor*. Table IX shows all parameters within all the descriptors involved in the *Clock Reference Stream* [44]. All the values listed in the table are set to zero, except *hasRandomAccessUnitsOnlyFlag*=1 and *objectTypeIndication*=0xFF.

TABLE IX: Configuration values from SL packet, DecoderConfigDescriptor and SLConfigDescritor when Clock Reference Stream in used [44]

Descriptor	Field
SL Packet	It shall not convey a SL packet payload
	The SL packet only conveys OCR values
Decoder Config	hasRandomAccessUnitsOnly Flag (value 1)
	objectTypeIndication (value 0xFF)
	bufferSizeDB
SL Config	useAccessUnitStart Flag
	useAccessUnitEnd Flag
	useRandomAccessPoint Flag
	usePadding Flag
	useTimeStamps Flag
	useIdle Flag
	duration Flag
	timeStampResolution
	timeStampLength
	AU_length
	degradationPriorityLength
	AU_seqNumLength

There are several constraints to be considered. All ESs with no OCR information require waiting until the ES conveying the OCR values is available. Once the ES with the OCR is available at the decoder all ESs with no OCR are synchronized to the other streams and, finally, if the ES with OCR is unavailable or it is modified, all ES depending on it are treated equally. Needless to say that if an ES without OCR suffers any alteration, it does not affect any of the other ES sharing the same time object.

4) M4Mux Tool: The M4Mux[14] is a tool used for the delivery of low bitrate and low delay streams, such as object descriptor and scene description. It contains interleaving SL-packetized streams with instantaneous bitrate. M4Mux packets have variable size and they convey one or multiple SL packets. Every SL packetized stream is assigned to an M4Mux channel [44].

M4Mux uses two operational modes: Simple mode and Muxcode mode. The former only conveys one single SL packet in each M4Mux packet, whereas the latter conveys multiple SL packets within one M4Mux packet [44].

The simple mode only adds two 8-bit fields in the M4Mux header: *index* and *length*. The Muxcode mode adds an additional third 4-bit *version* field. The M4Mux structure of the Simple and Muxcode modes are depicted in Fig. 14.

The *fmxClockReference* (FCR) format, which is the clock reference for M4Mux streams, is indicated at the M4Mux Timing Descriptor, depicted in Fig. 15. There are three fields related to M4Mux timing: 32-bit *FCRresolution* field, 8-bit *FCRlength* field and 8-bit *FmxRateLength* field. The clock references and rate are conveyed into the M4Mux Packet header within the *fmxClockReference* and *fmxRate* fields. The

[14]M4Mux is also known as FlexMux. The term FlexMux is used in MPEG-2 part 1 document and M4Mux in MPEG-4 part 1 document. In ISO/IEC JTC 1/SC 29/WG 11 N5677 document, it is stated that FlexMux is a copyrighted term, and, therefore, M4Mux should be used.

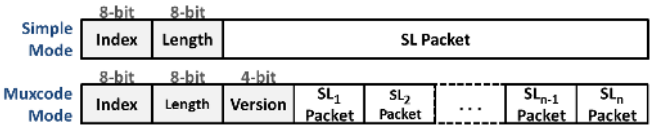


Fig. 14: M4Mux modes. High Level Packet Structure

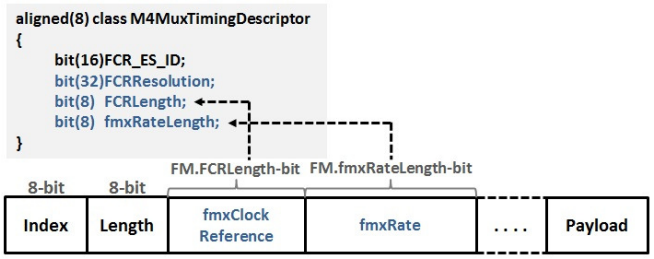


Fig. 15: High Level diagram M4Mux Timing Descriptor

arrival time of the byte i of the M4Mux stream can be calculated from *fmxClockReference* by using the following equation [44]:

$$t(i) = \left(\frac{FCR\,(i'')}{FCRres}\right) + \left(\frac{i - i''}{fmxRate(i)}\right) \qquad (26)$$

being i'' the byte index of the last *fmxClockReference* bit within the M4Mux stream, and i the *'the index of any byte within the M4Mux stream'* [44] where (i"<i); FCR (i") is *'the time encoded in fmxClockReference in units of FCRresolution'* [44] and, finally *fmxRate(i)* is *'the rate specified by the fmxRate field for byte i'* [44].

Table X summarizes all clock references and timestamps used in MPEG-1, MPEG-2 and MPEG-4.

F. ISO Base Media File Format (BMFF)

ISO Base Media File Format (BMFF) is a '*base format for media file formats*' [45] containing timing, structure and media information. It aims to be independent from network protocols. ISO BMFF, with MP2T media container, is one the formats used in MPEG-DASH (explained in next section) for media delivery.

ISO BMFF files are made of objects or boxes. All data within an ISO media file is inside a box. There are multiple boxes defined in [45], but only those relevant to timelines are presented in this paper. Boxes are defined using Syntax Description Language (SDL), defined in ISO/IEC 14772-1 [46]. In Fig. 16, a group of all the boxes defined in [45] is shown in order to provide a high level view of the ISO BMFF hierarchy.

ISO BMFF defines *brands*, which specify a subset of requirements to be met by an ISO base media file. An example of an ISO BMFF file used by the MS-SSTR protocol is found in Fig. 17. The ISO BMFF file in Fig. 17 is structured as follows: an initial File Type (*ftyp*) and Movie Metadata box (*moov*) followed by multiple Movie Fragments (*moof*) and Media Data (*mdat*) boxes. The Movie Fragment Random Access (*mfra*) box ends the media file.

In the following sub-sections the time information conveyed

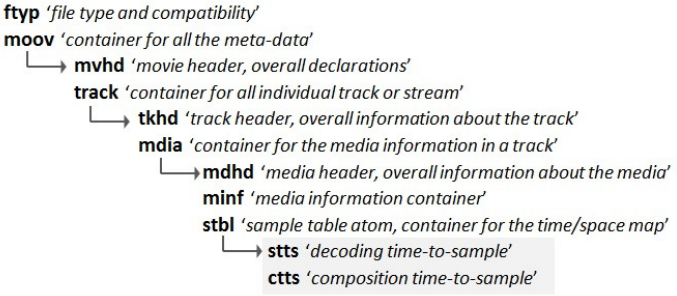


Fig. 16: ISO BMFF hierarchy for clock references and time-stamps related boxes [45]

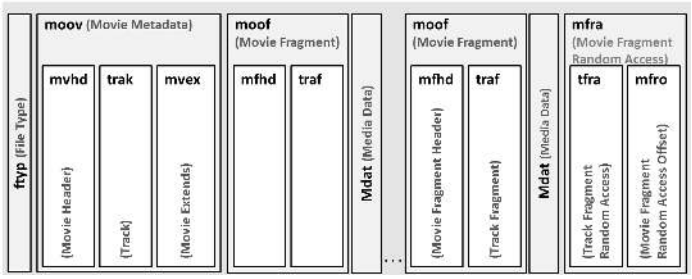


Fig. 17: Example ISO BMFF system used by MS-SSTR [47]

within the ISO BMFF file type, which differs from the time information conveyed in other MPEG standards, is explained.

1) ISO BMFF Time References: The clock references, as we have previously seen, are not present in ISO BMFF file type. Time information in ISO BMFF files is delivered once in each related box within the ISO file.

Time references are found in three different levels: movie, track and media, within their respective header's boxes. The boxes are Movie Header (mvhd), Track Header (tkhd) and Media Header (mdhd) boxes.

The Movie box is the '*container box whose sub-boxes define the metadata for a presentation*' [45]; the Track box conveys '*timed sequences of related samples in an ISO base media file*' [45], i.e., a sequence of images or audio samples; and finally, the Media Data box is the '*box which can hold the actual media data for a presentation*' [45], i.e., contains the media samples within a track. Therefore, *mvhd* conveys overall declarations, *tkhd* conveys track information and, *mdhd* conveys information about the media [45].

The time related fields that can be found in the three boxes' headers are: *creation_time*, *modification_time*, *timescale* and *duration*. All fields can be 32 or 64-bits, depending on version of the box used. Table XI summarizes the values of these fields in every box header.

2) ISO BMFF Timestamps: Timestamp related boxes are the Decoding Time (DT) to Sample Box (stts) and the Composition Time (CT) to Sample Box (ctts). Their parent box is the Sample Table Box (stbl) as can be seen in Fig. 16. The *stts* box is mandatory and a minimum of one is required, whereas the *ctts* box is required when decoding and composition times are not equal. ISO BMFF timestamps are only present in one

TABLE X: Summary Timestamps and Clock References in MPEG-1 (Section III-C), MPEG-2 (Section III-D) and MPEG-4 (Section III-E)

	Standard	Field	Resolution	Frequency	Periodicity	Location
Clock References	MPEG-1	SCR	33-bit	90kHz	0.7s	Pack Header
	MPEG-2 PS	SCR	42-bit	27MHz	0.7s	Pack Header
		ESCR	42-bit	27MHz	0.7s	PES Header
	MPEG-2 TS	PCR	42-bit	27MHz	0.1s	AF Header
		OPCR	42-bit	27MHz	-	AF Header
		ESCR	42-bit	27MHz	0.7s	PES Header
	MPEG-4 SL	OCR	SL.OCRlength (8-bit)	SL.OCRresolution (32-bit)	0.7s [21]	SL Header
	MPEG-4 M4Mux	FCR	FCRlength (8-bit)	FCRresolution (32-bit)	0.7s [21]	M4Mux Packet
Timestamps	MPEG-1	PTS	33-bit	90KHz	-	Packet Header
		DTS	33-bit	90KHz	-	Packet Header
	MPEG-2 PS	PTS	33-bit	90KHz	0.7s	PES Header
		DTS	33-bit	90KHz	-	PES Header
	MPEG-2 TS	PTS	33-bit	90KHz	0.7s	PES Header
		DTS	33-bit	90KHz	-	PES Header
		DTS_next_AU	33-bit	-	-	AF Header
	MPEG-4 SL	CTS	SL.TSlength (8-bit)	SL.TSresolution (32-bit)	-	SL Header
		DTS	SL.TSlength (8-bit)	SL.TSresolution (32-bit)	-	SL Header

level within the ISO box structure, within the *stbl* boxes [45].

The time related boxes contain information related to samples. A sample is defined in [45] as '*all the data associated with a single timestamp*'. A sample can be an individual video frame or a compressed section of audio.

In the *stts* box three 32-bit fields can be found: *entry_count*, *sample_count* and *sample_delta* fields. The *entry_count* is the number of entries of *stts* box, the *sample_delta* is the delta between two consecutive DT values. The *sample_count* is the number of samples with the same *sample_delta* [45] (See Table XII). The decoding time for the n^{th} sample within the *stts* box is:

$$DT(n+1) = DT(n) + stts(n) \tag{27}$$

being *n* the index sample, *stts(n)* the table entry for sample *n*, *DT(n+1)* the decoding time for sample *n+1* and *DT(n)* is the decoding time for sample *n* [45].

The *ctts* box indicates the difference between decoding and composition time, being always the latter greater than the former. As the *stts* box, the *ctts* box also contains different 32-bit fields: the *entry_count* field (which is the number of the box entries of the *ctts* box), *sample_count* (which is the number of consecutive samples with the same sample offset [45], as can be seen in Table XII) and the composition time for the n^{th} sample within the ctts box is:

$$CT(n) = DT(n) + ctts(n) \tag{28}$$

being *n* the index sample, *DT(n)* the decoding time for sample *n*, and *ctts(n)* the table entry for sample *n* [45].

G. MPEG-DASH

MPEG-DASH standard [4] is a client-driven multimedia delivery protocol for Dynamic Adaptive Streaming over HTTP. The main characteristic of MPEG-DASH is the adaptive media delivery according to the variable network conditions and/or client's requirements. Using DASH, the client dynamically selects the most suited media quality according to the estimated network conditions (connectivity, bandwidth ...), its hardware

TABLE XI: Time References within ISO BMFF

	creation time	**modification time**	**timescale**	**duration** (in timescale units)
Movie Header Box	Movie creation time	Movie modification time	Time units in a second	Movie presentation duration
Track Header Box	Track creation time	Track modification time	Time units in a second	Track presentation duration
Media Header Box	Media creation time (in a track)	Media modification time (in a track)	Time units in a second	Media presentation duration

TABLE XII: *TimeToSample* Box and *CompositionOffset* Box Classes [45]

```
aligned (8) class TimeToSampleBox extends FullBox (stts, version=0,0) {
    unsigned int(32) entry_count;
    int i;
    for (i=0; i < entry_count; i++ {
        unsigned int(32) sample_count;
        unsigned int(32) sample_delta;
    }
}
```

```
aligned (8) class CompositionOffsetBox extends FullBox (ctts, version=0,0) {
    unsigned int(32) entry_count;
    int i;
    for (i=0; i < entry_count; i++ {
        unsigned int(32) sample_count;
        unsigned int(32) sample_offset;
    }
}
```

and decoding and/or to its processing load capabilities.

MPEG-DASH delivers small chunks of media files stored in HTTP media servers. Two file formats are used to store these media segments: MP2T and ISO BMFF. Therefore, the timelines will be defined by the respective file format within the media segments.

MPEG-DASH includes XML and binary formats for HTTP servers/clients (complying with RFC2616 [48]) for media delivery. The Media Presentation Description (MPD) file is a key feature of MPEG-DASH. This file informs the client where and how to stream the media from the HTTP server. In the MPD file, some timelines are included within the different elements. The MPD file obeys the following pattern: within a unique MPD there are multiple Periods; and, inside every period various AdaptationSets may be found. Every AdaptationSet conveys a variable number of Representations. Finally, every Representation can convey multiple Segments (See Fig. 18).

Based on the MPD type, time restrictions and fields vary. An MPD can be either Static or Dynamic. Static MPD are generally used for stored media, while Dynamic MPDs are used for live media. We can see an example of MPD structure in Fig. 18 where the main elements of a Static MPD file delivering MP2T media Segments [49] can be found.

An example of the behaviour of a MPEG-DASH Client is shown in Fig. 19. The MPD file and the media Segments are stored in a HTTP media server. The client sends an HTTP request, so the server sends the MPD file. The client, once it has the MPD information, selects an AdaptationSet and one Representation. Then, it requests a list of media Segments for that selection, for every interval of time. The last step is to fetch the media Segments from the HTTP server [4].

Time related information can be found in the MPD file, and in Period and Segment elements. All of them follow either the format *xs:duration*, *xs:dateTime* or *xs:UnsignedInt* format [50].

Within the MPD file there are mandatory fields, such as *MinBufferTime*, *availabilityStartTime* (mandatory for Dynamic type), or *MediaPresentationDuration* (mandatory for Static type), and optional fields, such as *availabilityEndTime*, *minimumDatePeriod*, and *timeShiftBuffer*. Additional information is included in a Period element, such as *start* and *duration* of the period. Finally, in the Segment element time fields such as *timescale*, *presentationTimeOffset* and *duration* are included. In Table XIII all the time-related fields in MPEG-DASH are listed, including the field type and a brief description with values and restrictions.

A Period element represents the time frame of the media play-out. Information such as *start* and *duration* indicate the beginning and duration of the play-out of the Period element, respectively. If *start* is missing then the beginning of the Period element is the *start* plus the *duration* of the previous Period element. If the first Period in the MPD has no start information, then the MPD type is Static and *start* value is zero.

Segment elements provide information about the media location, availability, properties, and the timing information included within a Representation. There are four types of Segments: Initialization Segments (‘*Describes the Initialization Segment*’, Media Segments (‘*Describes the accessible Media Segments*’, Index Segment (‘*describes the accessible Index Segments*’) and Bitstream Switching Segments [4]. Each of them conveys the related information. In relation to MPEG-DASH timelines, the Index and Media Segments will be described.

The Index Segment (SIDX), defined in ISO BMFF file format, provides the index information to access the Media Representation. It contains the following time related fields 32-bit *timescale*, 32-bit *subsegment_duration* and *ear-*

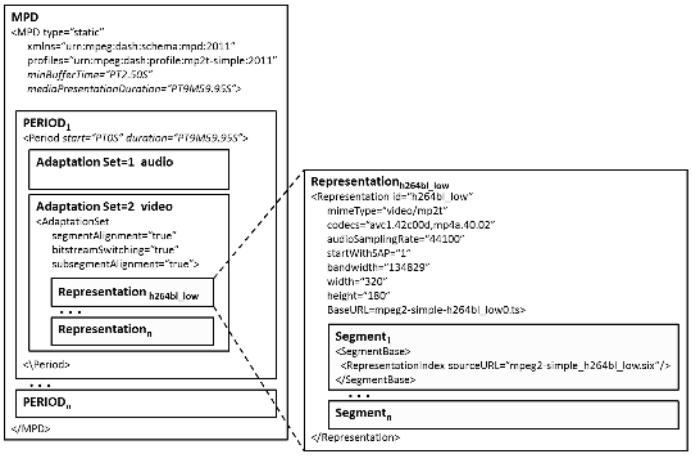


Fig. 18: MPD File Structure. Example of a Static MPD file type for MP2T Streaming [49]

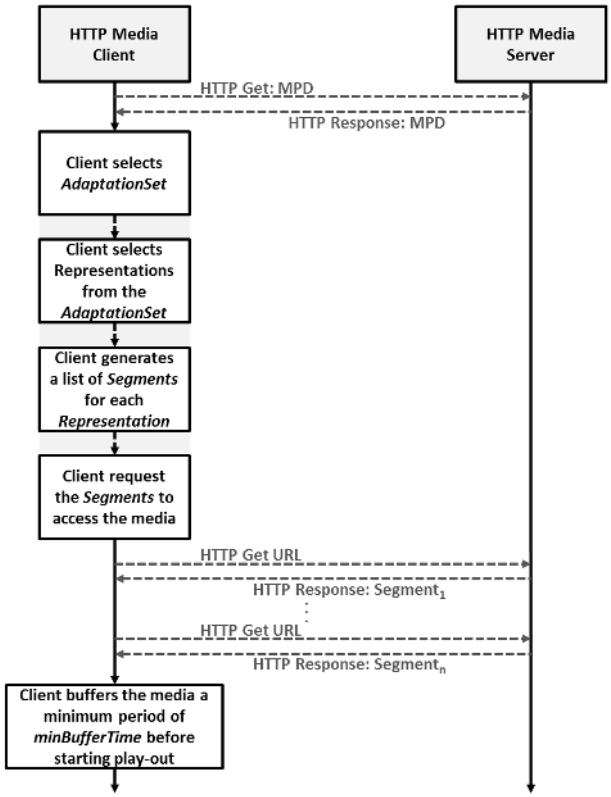


Fig. 19: High Level MPEG-DASH Client behaviour example from [4]

TABLE XIII: Time Fields in MPD, Period and Segment within the MPD File. A summary from [4]

Element	Field	Format	Description
MPD	*availabilityStartTime*	*xs:dateTime*	For Dynamic type it codes the earliest availability of all segments. For Static type it conveys the segment availability start time. If it is not present, segments availability is equal to the MPD availability.
	availabilityEndTime	*xs:dateTime*	Latest availability for all segments. The value is not set when *availabilityEndTime* tag is missing.
	mediaPresentationDuration	*xs:duration*	'Duration of the entire media Presentation' [4]. Its value is not known when not present but it is mandatory when the *minimumUpdatePeriod* field is found.
	minimumUpdatePeriod	*xs:duration*	The minimum period of time the MPD file can be modified. MPD is not modified when *minimumUpdatePeriod* tag is missing, and for type Static this field shall not be included
	minBufferTime	*xs:duration*	'Common duration used in the definition of the Representation data rate' [4]. Minimum length in time of media stored in buffer before the beginning of play-out.
	timeShiftBufferDepth	*xs:duration*	Time Shifting Buffer guaranteed. For type Dynamic when *timeShiftBufferDepth* tag is not included, the value is infinite. For type Static the value is not defined.
	suggestedPresentationDelay	*xs:duration*	For type Dynamic it indicates the fixed delay offset for the AUs presentation time. For type Static the value is not required and if present should be disregarded.
	maxSegmentDuration	*xs:duration*	It establishes the segments maximum duration within the MPD.
	maxSubsegmentDuration	*xs:duration*	It establishes the subsegments maximum duration within the MPD.
Period	*start*	*xs:duration*	It indicates the Period start time. It establishes the start time of each Period within the MPD and each AU presentation time in the Media Presentation timeline.
	duration	*xs:duration*	It indicates the Period time duration.
Segment	*timescale*	xs:unsignedInt	It represents the timescale in units per seconds.
	presentationTimeOffset		Presentation time offset related to the period's start. Default value is zero.
	duration	*xs:duration*	It conveys the Segment time duration.
	SegmentTimeline	—	It indicates the earliest presentation time and duration of segments within the Representation.

```
<Representation id = "1" mimeType = "video/mp2t"
    width = "320"
    height = "180"
    BaseURL = mpeg2-simple-h264_low.ts>
    <SegmentBase timescale = "1000"
        duration = "10000"
        startNumber = "0">
        <RepresentationIndex ="video320x180_48kbps_Index.didx">
        </RepresentationIndex>
    </SegmentBase>
</Representation>
```

Fig. 20: Example of *SegmentsBase* with time fields [49]

```
<Representation id = "1" mimeType = "video/mp2t"
    width = "320"
    height = "180"
    BaseURL = mpeg2-simple-h264_low.ts>
    <SegmentTemplate
        timescale = "1000"
        duration = "10000"
        startNumber = "0"
        media = "video320x180_48kbps$Number$.ts"/>
        <RepresentationIndex="video320x180_48kbps_Index.didx">
        </RepresentationIndex>
    </SegmentTemplate>
</Representation>
```

Fig. 21: Example of *SegmentTemplate* with time fields [49]

liest_presentation_time (32- or 64-bit field depending on the version). These fields establish restrictions within the *SegmentTimeline*, which will be detailed later in this section.

The Media Segments can be represented by three types of structures: *SegmentBase*, *SegmentTemplate* and *SegmentList*. *SegmentBase* is used to provide information for a single Media Segment. *SegmentTemplate* and *SegmentList* are used for multiple Segments information. An example of *SegmentBase* can be seen in Fig. 20, while an example of *SegmentTemplate* can be seen in Fig. 21.

There are two options to describe Segment timelines. First, time fields structure can be included within the Segment element. Second, *SegmentTimeline* can be added, which provides the means to signal arbitrary and accurate segment durations and to signal timeline discontinuities within the Media Presentation. It has three fields: duration (*d*), time (*t*) and repetition (*r*). *t* indicates the MPD starting time (default value is zero), *d* gives the Segment's duration and *r* the number of segments with the same *d* value. An example of the time fields within the *SegmentTemplate* can be seen in Fig. 21, while an example of a *SegmentTimeline* can be seen in Fig. 22.

```
<Representation id = "1" mimeType = "video/mp2t"
    width = "320"
    height = "180"
    BaseURL = mpeg2-simple-h264_low.ts>
    <SegmentTemplate
      timescale = "1000"
      startNumber = "0"
      media = "video320x180_48kbps$Number$.ts"/>
      <SegmentTimeline>
          <S t = "0" d = " " r = " " />
      </SegmentTimeline>
      <RepresentationIndex="video320x180_48kbps_Index.didx>
      </RepresentaionIndex>
    </SegmentTemplate>
</Representation>
```

Fig. 22: Example of *SegmentsTemplate* and *SegmentTimeline* [49]

The *SegmentTime* fields need to meet some requirements if *$time$* identifier is present within the *SegmentTemplate* representation. The *timescale* fields need to be identical in the Segment and the SIDX. The field *SegmentTimeline t* shall be equal to the *earliest_presentation_time* and the field

SegmentTimeline d equal to the *subsegment_duration*.

SIDXs include a Segment to provide MP2T PCR information named *MPEG2TSPCRInfoBox*, which maps the PCR value of the '*first sync byte of the first MP2T packet in the media Subsegment*' [4]. This value could be different from the PCR value of the first MP2T packet within the segment because this relates to the last bit of the PCR_{base} [4].

The media content specified in MPEG-DASH, MP2T and ISO BMFF, within the MPD segments shall comply with some requirements. As an example, for MP2T streams, Media Segments shall contain full PES packets within the MP2Ts, and Media Segments shall only convey one single Program. Initialization information shall be included within the Media Segment. Also, if the Index Segment is present, it shall convey all time-varying initialization information. Media Segments cannot rely on Initialization Information from previous Media Segments [4].

Play-out at receiver-side will not begin until the minimum required media is buffered, which means that the *minBufferTime* has been reached [4].

Once the initial play-out begins, the client will adaptively fetch media Segments taking into account in each moment the estimated network conditions and the available hardware and processing resources, also based on the MPD specifications. However, note that the specific switching strategy to be used is not specific in the standard.

An example of the use of MPEG-DASH can be seen in [51]. In this work, a new High Efficiency Video Coding (HEVC) MPEG-DASH data set for streaming from High Definition (HD) to Ultra High Definition (UHD) at different encoding bitrates and different encapsulation options is defined, '*the first data set mixing different temporal resolutions and bit depth per videos, with multiple adaptation paths*' [51].

The media segments have the same duration (2, 4, 6, 10 or 20s) and each begins with an Instantaneous Decoder Refresh (IDR) slice[15]. Different media bitrates are used and different methods of encapsulation tested are live profile, live profile with bitstream switching, on-demand and main profile.

MPEG-DASH is also used in [52] to provide '*flexible web-based access of video from sensors and other miniaturized source nodes*' proposing a Video Sensor Network Platform compatible with MPEG-DASH (WVSNP-DASH) using the HTML5 File System, where video segments are fetched, for video buffering and playback providing wide cross-platform support.

H. MPEG Media Transport (MMT)

Recently, MPEG has published the MPEG Media Transport (MMT) standard [9] to respond to the requirements of the new media consumption paradigm, where content can be accessed anywhere through heterogeneous scenarios and in a large variety of devices. Next-generation broadcasting systems will not work as independent content delivery systems, but as a part of a content delivery system using broadband networks. In addition, content-centric networking promises more efficient distribution of data through in-network caching and the propagation of content through the network. That use of both broadcast and broadband networks has to be transparent to the end users who can make use of content without being aware of the used delivery systems.

Moreover, MP2T-based delivery systems have some limitations regarding some issues, such as SVC, Multi-view Video Coding (MVC) delivery on more than one delivery channel, UHD TV delivery, etc [53].

MMT is being standardized as Part 1 of ISO/IEC 23008 [9], a new standard suite including HEVC and 3D Audio. It will be used for efficient and effective server-driven delivery of encoded media, including both timed and non-timed data over heterogeneous networks[16]. It aims to unify a media delivery protocol for broadcast and broadband delivery systems. Therefore, it incorporates the functions defined in previous MPEG standards, including ES structural relationships and synchronized play-out of media content, plus the required information for delivery-layer processing [54]. The general requirements for MMT are: adaptable and dynamic media components access; easy media format conversion between media storage and delivery; and the capability to use multiple multimedia components [55].

MMT defines formats and protocols categorized into three functional areas: encapsulation, delivery, and signalling. The Encapsulation Layer (MMT E-Layer) specifies the encapsulation format of encoded media data to be either stored or delivered. The Delivery Layer (MMT D-Layer) specifies the application layer protocol and the necessary payload format for exchanging encapsulated media data between network entities. The Signalling Layer (MMT S-Layer) specifies the format of signalling messages necessary to manage delivery and consumption of the media data [56]. The MMT architecture is shown in Fig. 23.

Unlike previous MPEG multimedia delivery technologies, focused on representing structural relationships of ES (such as MPEG-2 PSI) and carrying information for synchronized multimedia play-back, the content model of MMT focuses on providing the necessary information so the media data type and the delivery protocol are independent from the delivery layer [54]. Additionally, content model of MMT provides solutions to encapsulate non-timed media data (e.g., files or images) which are not associated with designated presentation time at the time of delivery, whereas the former technologies have been focused on the delivery of timed media data composed of series of AUs associated with designated presentation times at the time of delivery.

Fig. 24 shows the protocol stack of MMT, in which the scope of the MMT specification is shadowed (in grey). MMT model specifies the MMT packet and the payload format for delivery, and an MMT package as the logical structure of the content.

An MMT packet is a variable-length packet containing one MMT payload, which in turns contains just one kind of media data (it cannot contain different types of data or signalling

[15]A particular I-slice which signals the beginning of a GOP/sequence

[16]Data formatted according to the MMT specifications can be delivered by any packet-based network without using IP, such as Generic Stream Encapsulation (GSE) protocol, defined by DVB.

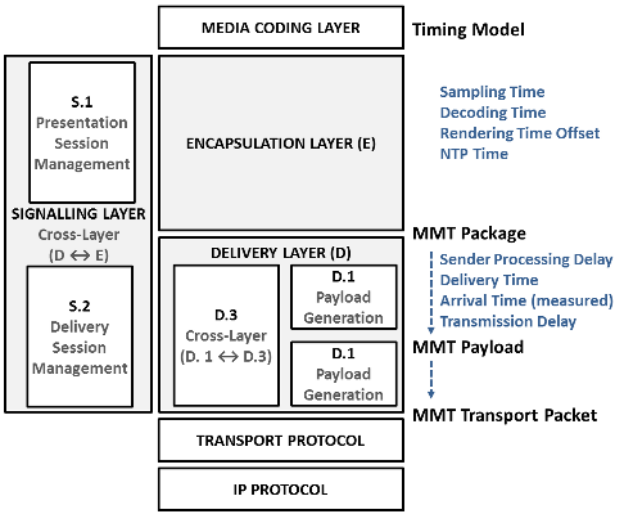


Fig. 23: MMT Architecture and functionalities with Timing Model proposed in [56]

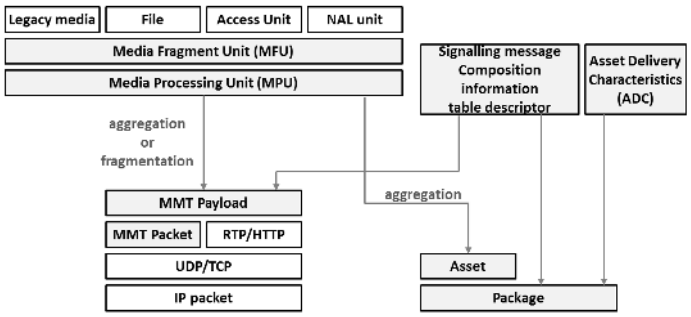


Fig. 24: MMT protocol stack [53]

messages which can be transferred in MMT packets in one IP data flow).

An MMT package is a logical entity including encoded media data about the content, called MMT assets, and information for the delivery-layer processing, such as Composition Information (CI) and Asset Delivery Characteristics (ADCs). An MMT package carries one CI and one or more ADCs.

An asset defines the logical structure carrying encoded media data. Any type of data that can be individually consumed is considered a separate asset. An asset encapsulates encoded media data such as audio, video or a web page data of timed or non-timed nature. Examples of data types that can be considered as individual assets are an MP2T file, MP4 file, or a JPEG file. An asset collectively references a number of Media Processing Units (MPUs) with the same Asset ID (a globally unique identifier used to refer to an asset). This allows the MMT package to be easily constructed by logically referring to MMT assets by their identifiers without specifying their physical location or physically embedding them.

An MPU contains at least one AU for timed data or partial data from a non-timed MMT asset. Due to the possible constraints of the underlying delivery networks related to its allowed maximum transfer unit, MPUs include small fragments of the data, known as Media Fragment Units (MFUs). This enables the dynamic adaptive packetization of the MPU during the delivery process. MFUs include fragments of encoded media data which can be independently decoded or discarded (e.g., a unit of an AVC bitstream). The MMT standard designed the MPU and MFU structures as a common data unit for both storage and packetized delivery of an MMT package. A system can easily and efficiently convert an MMT file to MMT packets, by processing the MPU headers and packetize it at the MFU boundaries when necessary (and vice versa). Fig. 25 shows all the above relations.

Regarding timing, the MMT model shall support media sync plus delivery-media processing functions. MMT's CI specifies the spatial and temporal relationships among the MMT assets (useful to determine the assets delivery order). It also provides information for associating assets to a specific screen (e.g., for multiscreen applications). This information can be useful to determine delivery configuration in heterogeneous delivery scenarios.

The presentation time of the first AU positioned in a MPU of the asset is described in signalling messages in order to synchronize the presentation of the media components. The presentation duration of each AU in one MPU is described in the MPU header. A receiver terminal identifies MPUs constituting the content and the presentation times (of each AU) by processing signalling messages. In MMT the presentation time is described on the basis of UTC. Therefore, the receiver can consume MPUs in a synchronized manner even if they are delivered on different channels from different sources.

On the one hand, the MMT D-Layer functions should include the capability to calculate delivery timing information, such as network delay, and the means to re-adjust timing relationships based on compensating the network jitter [56]. On the other hand, the MMT E-Layer should provide the timing information required for the correct media play-back at receiver-side and the delivery time, based on the temporal requirements. The features should include the conversion between MPEG transport/storage formats and MMT, and vice versa [56].

It is specified in [57] that every element in the delivery path is UTC synchronized (e.g., via NTP or other clock sync technologies). The principal benefit is that all media sources and end-users have access to a common (or related) global clock reference, although, adding in-line clock references would cause MMT to become more widely deployable.

In [56] the MMT timing system is presented. This system is intended to facilitate media sync in an MMT based media service. It proposes a timestamp-related header format for MMT E- and D-Layer timing models providing the tools for the sender/receiver sync media from several media sources.

The sampling time is a obtained from a 90KHz resolution clock that becomes fully compatible with DTS and PTS values in MP2T. Also, the sampling time structure, called *sampling_time_base*, fully follows the DTS and PTS bit size. The advantage of this system is that MMT E-Layer additionally includes an NTP time to link the sampling time with the UTC time [56].

Fig. 25 and 26 present the time model within the MMT

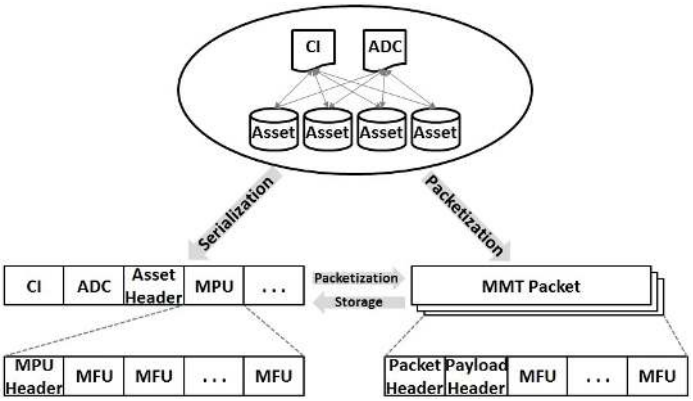


Fig. 25: Relationship of an MMT package's storage and packetized delivery formats [54]

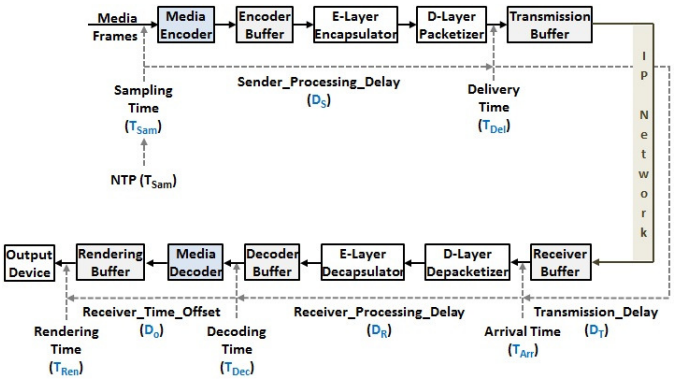


Fig. 26: MMT model diagram at MMT sender and receiver sides. Fig. 3 and 4 from [56]

architecture main layers. Next, the important instants in the MMT E-Layer are listed.

On the one hand, the list of timestamps in the MMT E-Layer is the following [56]:

- *Sampling Time* (T_{sam}): It is the sampling time of the first AU within a MPU. Timestamp reflecting the '*sampling instant of the input frame to the media encoder*' [56].
- *Decoding Time* (T_{dec}): It is the decoding time of the first AU within a MPU. Timestamp reflecting the decoding instant of the input frame to the media encoder.
- *Rendering Time* (T_{ren}): It indicates the MDU presentation/composition time after rendering time offset.
- *Rendering Time Offset* (D_o): Timestamp indicating the time in rendering buffer to reorder and decode media frames ready for presentation. It is the time difference between decoding and presentation time.
- *NTP Time*: Timestamp representing the sampling time with a UTC time in NTP-based format.

The values of *sampling_time*, *decoding_time*, *rendering_time_offset* and *NTP_time* are established through the media encoding and encapsulation stages, and are included as timestamps in the MMT packets and files.

On the other hand, the list of important instants in the MMT D-Layer is the following [56]:

- *Delivery Time* (T_{del}): It is the measured time of the MMT packet to be delivered after being processed by the sender, and ready for the transmission over the IP network. It is the elapsed time needed from the *sampling_time* (T_{sam}) until the MMT is ready to be sent to the transmission buffer.
- *Arrival Time* (T_{arr}): It is the measured time of the MMT packet arrival at receiver-side. It represents the transmitted MMT packet arrival time at the receiver.
- *Sender Processing Delay* (D_s): Timestamp specifying the elapsed time from the moment an MDU enters into the media decoder until an MMT packet ready for delivery is generated.
- *Transmission Delay* (D_t): It is the time elapsed from the delivery time (T_{del}) until the arrival time (T_{arr})

The MMT E-Layer timing model provides timing pairs of *sampling_time* and *NTP_time* fields. Thus, the *sampling_time* is mapped to a wall-clock time providing an universal timebase among multiple streams, from different sources, to synchronize, at the receiver, the decoding time of media packets [56].

MMT timestamps are UTC based, whereas MP2T timestamps are STC based. Accordingly, in order to synchronize these different types of timestamps in MMT and MP2T, additional messages are needed. These are called Clock Relation Information (CRI) messages [54]. They include a CRI Table providing the mapping time information between the UTC clock (e.g., an NTP Clock) and MPEG-2 STC. These messages are necessary to inform such relationship to an MMT receiving entity by periodically delivering values of the UTC and the STC times at the same time instants. If more than one MPEG-2 ES with different MPEG-2 STCs are used, more than one CRI descriptor[17] are delivered. This is an additional tool to sync media presentation at end-user in hybrid delivery systems. At an MMT receiver the MP2T's STC is linked to an UTC wall-clock value via the information provided by the CRI descriptor.

On the other hand, actual media transport protocols, such as RTP, shall be supported by MMT. In [56], fully compatible MMT and RTP timelines are proposed. To keep compatibility with 32-bit RTP timestamp, two fields are used to represent the sampling time: 1-bit *sampling_time_ext* and 32-bit *sampling_time_base*. The *sampling_time_base* field in the MMT E-Layer timing information should be associated to the 32-bit RTP timestamp field in the RTP header.

Next, a brief comparison between MMT, RTP and MP2T is provided. MP2T is the current technology for broadcasting systems but does not provide features for hybrid delivery. MP2T does not provide non-real-time content, due to the difficulty of delivering content as a file. An added drawback is that the STC is not shared between encoders. As a result from the media sync perspective, it is required to sync the STC of multiple servers.

RTP delivers individual media components. It supports multiplexing media components with signalling messages, but it does not assist content file delivery. Therefore, a content component cannot be delivered as a file. An added drawback is that no storage format is specified by RTP.

[17]A CRI descriptor is used to specify the relationship between the NTP timestamp and the MPEG-2 STC. It is carried in the CRI Table

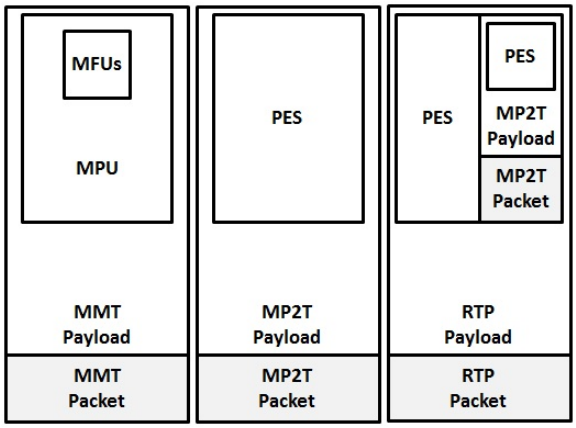


Fig. 27: Comparison of protocol layer structure of MMT, MP2T and RTP

In [53] a functional comparison between MMT, MP2T and RTP is presented (See Table XIV and Fig. 27).

MMT protocol aims to provide all MP2T and RTP missing features to facilitate Next Generation Networks (NGN) broadcasting systems [53].

The MMT approach provides the additional functionality of QoS management of media assets, as well as of multiplexing several media components into a single flow. MMT includes the following delivery functionalities: media sync based on UTC, multiplexing media assets into a single or multiple flows, and buffer management.

TABLE XIV: Functional comparison of MMT, MP2T and RTP [53]

Function	MMT	MP2T	RTP
File Delivery	Yes	Partially yes	External
Multiplexing media components and signalling messages	Yes	Yes	No
No multiplexing media components and signalling messages	Yes	No	Yes
Combination of media components on other networks	Yes	No	Yes
Error resiliency	Yes	No	External
Storage format	Partially yes	Partially yes	No

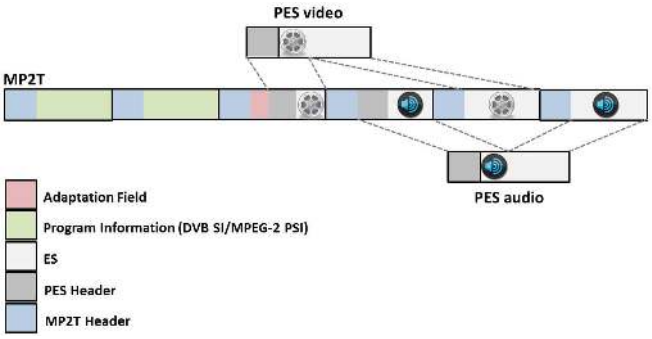


Fig. 28: DVB/MPEG-2 Stream Packets distribution

IV. Time and Timing within DVB Systems

A general overview of the DVB project and the development of technical specifications for DVB is presented in [58]. On the technical side, a high level description of the delivery of DVB services over the Internet is presented in [59]. In [60] the guidelines to use audio and visual coding in broadcast technologies are described. This section is focused on the DVB SI tables used to transmit services, programs, events and application information, and more specifically, on the tables used to transmit time within the DVB stream.

DVB streams utilize MP2T for media streams delivery. Within DVB systems, time/timing information is shared via information tables, where every table is conveyed within MP2T packets. DVB uses DVB SI tables and MP2T employs MPEG-2 PSI tables. There is a tight relationship between both systems' tables to provide all the information needed by the decoder to achieve media sync.

Fig. 28 describes the high level packet distribution within a DVB/MPEG-2 stream. At the beginning of the stream, packets containing DVB SI and MPEG-2 PSI tables carrying program information can be found (also inserted periodically along the stream) and then multiple MP2T packets containing PES of different media types used in the MP2T stream. Adaptation field is inserted when clock references need to be encoded and PES headers would be inserted at the beginning of every PES.

The complete structure of the Information Tables with the name of each table is shown in Fig. 29. DVB streams deliver services and each service has multiple programs. The system to link to each other is via the Service Description Table (SDT), from the DVB SI, and the Program Association Table (PAT), from the MPEG-2 PSI. The PAT contains the connection between a program and a DVB Service. Every program in the PAT is linked to a service in the SDT via the *transport_stream_id* (16-bit). Moreover, the PAT is linked to the Program Map Table (PMT) via the *program_number* (16-bit). Therefore, PAT connects SDT (in MPEG-2 PSI) with PMT (in DVB SI).

In the DVB SI, the time related tables are the Event Information Table (EIT), the Time and Date Table (TDT) and the Time Offset Table (TOT). Briefly, EIT provides programs' initial play-out time; TDT includes UTC time information; and, finally, TOT adds the local time offset of the geographical media delivery region. TDT and TOT are used to deliver the time to the end-users' devices [61], both conveying the 40-bit *UTC_time* field signalling the current time in UTC, using Modified Julian Date (MJD) format [61].

TOT conveys the *local_time_offset_descriptor* which informs of the 24-bit *country_code* field, 6-bit *country_region_id* field, the 1-bit *local_time_offset_polarity* field and the 16-bit *local_time_offset* field.

There are two types of EIT: *event schedule* and *present/following event information* tables. Both are used to inform of the service events within the DVB stream. The EIT *event schedule* table contains multiple events, whereas the *present/following event* table only informs of the present and following event. The presence of EIT table is indicated in SDT by means of

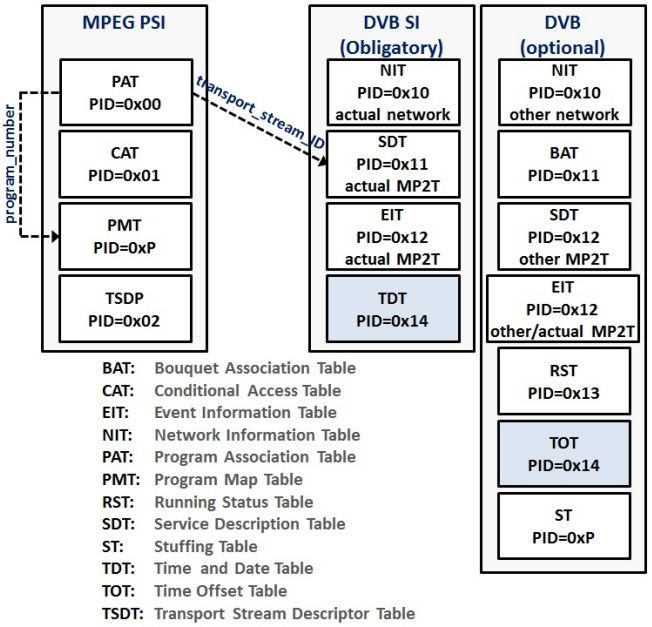


Fig. 29: High Level DVB SI and MPEG-2 PSI tables [61]. In blue time related tables

the *EIT_schedule_flag* (1-bit) and *EIT_present_following_flag* (1-bit) fields [61].

The EIT informs of the initial play-out time of a particular event within a service via the 40-bit *start_time* field, which contains the UTC time in MJD format of the play-out starting time. The 24-bit *duration* field, in EIT, informs of the time duration of the event in hours/minutes/seconds. This information creates the link between the wall-clock time of an event initial play-out time and the MP2T program.

The constraints to send these information tables are not very tight. TDT and TOT tables must be sent within 25ms and 30s threshold, whereas the EIT/SDT tables must be sent within 25ms and 2/10s threshold, depending on whether it refers to EIT/SDT for the actual MP2T or for other MP2T [21] [62] [63]. A recent study of real DVB-T multiplexed streams [64] showed a constant 25s gap between consecutive TDTs (value within the standard threshold). The same work detected time differences between PCR and TDT values up to 2s.

As well as the time related DVB SI tables, the DVB standards propose a specification to convey synchronized auxiliary data in DVB TS via the addition of a synchronized auxiliary data stream with included descriptors to facilitate media sync, which is explained below.

A. ETSI 102 823: Carriage of Synchronized auxiliary data in DVB TS

The ISO/IEC 13818-1 specification [21] describes how all the ESs of a service need to be encoded following specific timing model rules, in order to guarantee media sync at the receiver-side. It is the technique used in [36], [65], [66], [67], [68] and [69].

In ETSI 102 823 a generic tool to convey sync auxiliary data within DVB streams conveyed in MP2T/PES packets is specified. It uses multiple descriptors to insert a broadcast timeline which facilitates the auxiliary data synchronization with other ES within the same DVB service.

If included in a DVB Service, the PMT table of the MPEG-2 PSI for that service includes the synchronized auxiliary data ES PID (Packet Identifier) to associate it to the DVB Service. PES *stream_type*=0x06 and *stream_id*=0xBD identify the synchronized auxiliary data ES.

A DVB service could carry multiple ESs conveying synchronized auxiliary data, but every PES header is linked to an individual PTS value.

Different types of descriptors (to be included in the payload of the auxiliary data structure) are defined in [35] which are used to sync DVB auxiliary data to a broadcast timeline:

- *TVA_id* descriptor: It is used to enumerate the *TVA_id*s (TV-Anytime event identifier) and its state. It shall be repeated at least once every 2s.
- *Broadcast_timeline* descriptor: It describes the broadcast timeline used by the *time_base* descriptor. It provides the tool to map a time value with a particular point in the broadcast stream.
- *Time_base_mapping* descriptor: It is the tool to map an external time base with a broadcast timeline. The descriptor shall be transmitted at least once every 5s.
- *Content_labelling* descriptor: It labels an item of DVB content to facilitate metadata to reference a specific content. It provides the tool to map a broadcast timeline with the content item. This descriptor shall also be transmitted at least once every 5s.
- *Synchronized_event* descriptor: It conveys the information of an application-specific event to be synchronized with other components of the broadcast stream.
- *Synchronized_event_cancel* descriptor: It provides the tool to cancel a pre-defined *synchronized_event* descriptor that has not been reached in the broadcast stream.

The insertion of timing information within additional MP2T packets in a DVB stream, (conveying synchronized auxiliary data with absolute event timelines), provides a useful tool to facilitate media sync. This is the solution proposed by the HbbNext EU project ([36], [68], [69], [70]).

The main purpose of the system in [68] and in [69] is to facilitate a tool to synchronize third party broadband content to broadcast content by providing absolute time code (linked to the play-out time) within a DVB stream. Therefore, any broadband content could synchronized with the DVB stream by using these absolute references.

It provides frame-accurate sync owing to the fact that the absolute time code is related to a play-out time within the broadcast stream via PTS values. An absolute time code, using an MP2T packet, is inserted for every I-frame. Fig. 30 shows how the timecode within the auxiliary data PES is linked to an I-frame, containing a PTS, in the MP2T stream.

This solution requires firstly, the insertion of the timeline in the DVB stream at the broadcast media server, and secondly, a timeline extraction component at the receiver-side which also provides the sync component. The complete evaluation of this (hybrid) media sync strategy can be found in [70].

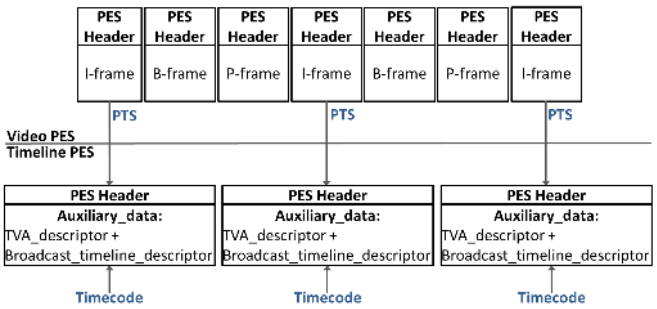


Fig. 30: MP2T timeline generation process [68] [69]

B. Delivery of Timeline for External Data

The proposed amendment to ISO/IEC 13818-1:2013 describes a method to map an MP2T program to embedded timelines. This method '*enables transport of a media timeline in an MPEG-2 TS program, in order to provide a stable media timeline not sensitive to PCR discontinuities*' [71]. Moreover, it provides a tool to signal the location of external media enhancements and the '*signalling of prefetching events*' [71]. It is achieved by including extra descriptors in the AF or by adding an extra program stream containing timeline descriptors. The Timeline and External Media Information (TEMI) describes external data and associated timing via descriptors.

There are two different techniques to include timeline information via descriptors. The first one is to include the descriptors in the AF (*af_descriptors*). The second one is to include *af_descriptors* in a program stream within the PES packets (see Fig. 31).

The addition of *af_descriptors* within the AF has the advantage of including the TIME information/descriptors with the minimum payload when bandwidth restrictions apply. The drawback is that AF size should remain small. Therefore when *af_descriptors* size is significant then the *af_descriptors* must be sent using a dedicated program stream within its PES packets.

To accomplish the first technique, the addition of one field, *af_descriptor_no_present_flag* in the AF is proposed. If this flag equals zero, then a list of *af_descriptors* is included in the AF (see left MP2T packet structure in Fig. 31).

To accomplish the second method, including *af_descriptors* within PESs, the program should be properly defined within the PMT table with the correct stream type. This stream conveys TEMI Access units (TEMI_AU), one in each PES packet. Every TEMI_AU payload may convey one or more *af_descriptors*, therefore they are Random Access Points [18] within the MP2T stream (see right MP2T packet structure in Fig. 31).

This program stream, like any other, is defined in the PMT table of the program (see Section IV). TEMI stream is signalled by *stream_type*=0x26 using the *private_stream_1* [19],[20] syntax. The TEMI_AU is conveyed within a PES packet which should have a PTS to link the presentation time to the time fields within the descriptors in the TEMI_AU.

There are three descriptors within TEMI: the *temi_location_descriptor*, *temi_base_url_descriptor* and *temi_timeline_descriptor*. The *temi_Location_descriptor* '*is used to signal the location of external data that can be synchronized with the program. It conveys several locations and their type (optionally including MIME types), along with the ability to signal upcoming external data association through a countdown until activation of the external data*' [71]. The *temi_base_url_descriptor* '*is used to assign a default base URL to all location descriptors*' [71]. Third, the *temi_timeline_descriptor* '*is used to carry timing information that can be used to synchronize external data. When the descriptor is carried within a TEMI access unit, the included timing information is given for the PTS value of the TEMI access unit carrying the descriptor*' [71].

The *temi_timeline_descriptor* is the descriptor which conveys the time information to link the PTS within the PES header to the timeline. The *temi_timeline_descriptors* is the means to link the PES PTS value to an NTP or PTP value, or to a media timestamp (MTP) described later, because the MP2T packet conveying the *temi_timeline_descriptor* with the indicated values also includes a PTS value in the PES header.

It has some related flags which inform about the presence of different time values, such as *has_timestamp* (2-bit), *has_ntp* (1-bit), *has_ptp* (1-bit), *has_timecode* (2-bit). The related fields are *media_timestamp* (32-bit), *ntp_timestamp* (64-bit) and *ptp_timestamp* (64-bit). The fields *frames_per_tc_seconds* (15-bit), *duration* (16-bit), *long_time_code* (64-bit) and *short_time_code* (24-bit) are only present if the *has_timecode* indicates that while also indicating if it uses the fields short or long time code.

The wall-clock time is conveyed by *ntp_timestamp* and *ptp_timestamp* fields, which will relate the PTS value of the PES header to an NTP or PTP timestamp (UTC time). Two subsequent PES PTS values are mapped via the following equations until a TEMI_AU, within a PES packet, is received:

$$NTP_i = \frac{(PTS_i - PTS_0)}{90000} + NTP_0 \tag{29}$$

$$PTP_i = \frac{(PTS_i - PTS_0)}{90000} + PTP_0 \tag{30}$$

In the previous equation, NTP_i indicates an *ntp_timestamp*, whereas the PTP_i is a *ptp_timestamp* value. *i* is the index of the *ntp_timestamp* or *ptp_timestamp*.

The *media_timestamp* (MTP)'*indicates the media time in timescale units corresponding to the PES PTS value of this packet for the timeline identified by the last temi_location_descriptor received*' [71]. Two subsequent PES PTS values are mapped via the following equations until a

[18] '*The process of beginning to read and decode the encoded bitstream at an arbitrary point*' [21])

[19] '*Private data is any user data which is not coded according to a standard specified by ITU-T — ISO/IEC and referred to in this Specification*' [21]

[20] It '*refers to private data within PES packets which follow the PES_packet() syntax such that all fields up to and including, but not limited to, PES_header_data_length are present*' [21]

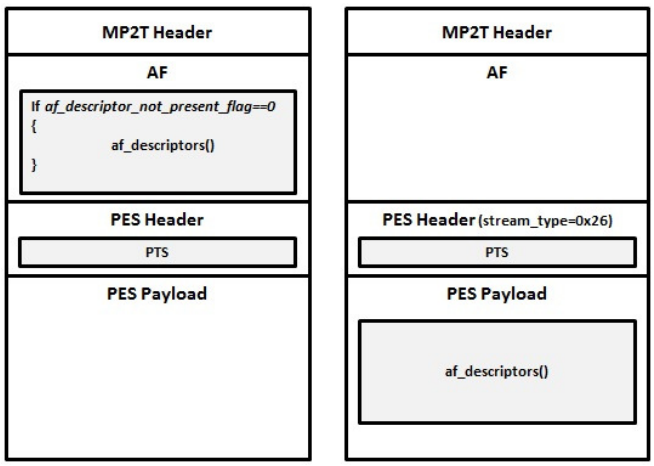


Fig. 31: MP2T packet structure with both TEMI

TEMI_AU, within a PES packet is received, timescale being '*the timescale used to express the media_timestamp*' [71]:

$$MTP_i = \frac{(PTS_i - PTS_0)}{90000} + \frac{MTP_0}{timescale} \qquad (31)$$

where MTP_i indicates a *media_timestamp* with *i* index.

V. RTP/RTCP

RTP and RTCP are transport layer protocols, specified in Request for Comments (RFC) 3550 [3], highly recommended for transmitting time-sensitive media data over IP networks.

RTP is used for media delivery, while RTCP is used to exchange valuable information about QoS metrics (e.g., jitter, packet loss), participants identification and media sync.

Multiple reports, such as [72], [73] and [74], argue that, although RTP is not compulsory for DVB-IPTV, the use of RTP for media delivery can provide many advantages, with the only minor drawback of adding a slight traffic overhead (due to the RTP header).

RTP typically runs on top of UDP, either in a unicast or multicast way, even though there is no restriction to use RTP on top of TCP. Each RTP packet can contain a fragment of one, or multiple AUs, and it includes in its header four valuable fields for media sync [3]: Synchronization Source (SSRC) identifier, sequence number, (generation) timestamp, and payload type. First, the SSRC identifier field (32-bit) allows for uniquely identifying RTP sources within a media session. Second, the sequence number field (16-bit) is used to detect packet loss and reconstruct the original order of incoming RTP packets at the receiver-side (as RTP does not guarantee ordered packet delivery). Third, the RTP timestamp (32-bit) is used to reconstruct the original timing for each RTP stream at the receiver-side. It is commonly derived from a local clock that must increase in a linear and monotonic fashion, producing a single and independent timeline for each RTP stream. Fourth, the payload type field (8-bit) gives information about the type of data conveyed within the RTP packet, the encoding mechanism being used, and the clock rate of RTP timestamps.

Multiple RFCs have specified new RTP payload types and payload formats for many media encoding mechanisms. The clear advantage of defining a specific RTP payload for each media is to provide as much compatibility as possible between different media formats conveyed via RTP, and treat them in a unified way. To define the RTP payload, three important issues shall be disclosed: first, the semantics of the RTP header; second, clear fragmentation norms; and, third, the procedure to associate media (e.g., video/audio) data to the RTP packets [75]. Some of the most relevant RFCs regarding to the specification of RTP payload formats for MPEG standards are the following: RTP payload format for MPEG-1/MPEG-2 video (RFC 2250) [76]; RTP payload format for MPEG-4 audio/visual streams (RFC 3016) [75]; RTP payload for transport of MPEG-4 elementary streams (RFC 3640) [77]; RTP payload format for H.263 Video Streams (RFC 2190) [78], RTP payload format for H.264 (RFC 6184) [79], and RTP payload format for SVC (RFC 6190) [80].

Fig. 32 illustrates the encapsulation of MPEG data (e.g., MPEG-2 and MPEG-4 payload) within an RTP packet, as well as the linking between RTP timestamps with MPEG time-stamps and clock references and with (wall-clock) NTP-based timestamps included in RTCP Sender Reports (RTCP SRs). RTCP SRs are regularly sent by media sources and, among other useful statistics, they convey a correspondence between RTP timestamps (32-bit) (obtained from a local clock) and NTP-based timestamps (64-bit) (obtained from a global clock, e.g., provided by NTP). On the one hand, this mapping time information will allow to check for, and correct, any inconsis-tencies between the local clocks of the sender and receivers, thus improving the intra-media sync performance. On the other hand, it will allow aligning the involved RTP streams in the time domain at the receiver-side, thus enabling inter-media sync. This is because the independent local timelines of each RTP stream can be mapped to the global reference wall-clock time. Moreover, the RTCP Source Description (RTCP SDES) packets [3] are also necessary to achieve inter-media sync. RTCP SDES packets can include a list of items conveying users' information (name, telephone, location). In particular, the CNAME (canonical name) item is used to associate the SSRC identifiers of each RTP stream (which are randomly generated) with a unequivocal and persistent identifier (in the form of *user@domain*) that will be shared by all the RTP streams to be synchronized.

In [81] and [82], the transport of MPEG media streams using RTP/UDP/IP protocols is described. The former explains the MPEG-2 delivery, whereas the latter describes the MPEG-4 delivery. The benefits provided by the use of RTP/RTCP in both cases are emphasized in these works. In addition, the work in [82] highlights the benefits of defining specific RTP payloads to convey concrete media types.

In RFC 7272 [83], RTP/RTCP protocols are extended to achieve IDMS, by defining two new RTCP messages. Finally, an overview of the capabilities of these protocols to provide (the different forms of) media sync and a discussion about the need of further work on this area is provided in [84].

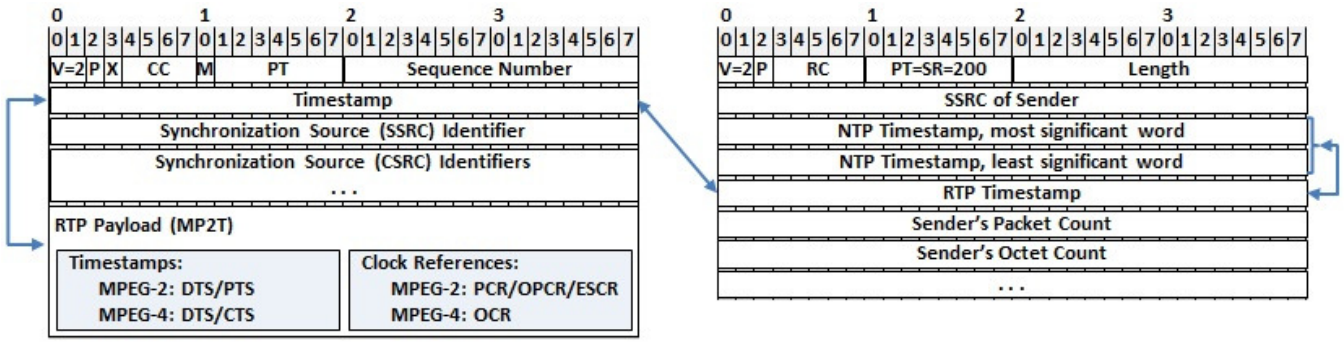


Fig. 32: RTP (left) and RTCP SR packets (right). RTP timestamp link with NTP timestamp and the RTP timestamp and MPEG payload including timestamps and clock references [3]

VI. Related Work

In this section, we compile various proposed solutions to enable synchronized media services by using any of the delivery technologies described in Sections III, IV and V. In Table XV a summary of references is listed by area.

In [85], the RTCP-based IDMS solution specified in RFC7272 [83] is implemented and evaluated, by using different architectural schemes, control algorithms and adjustment techniques.

In [86], the addition of a MediaSync module at the client is proposed to facilitate media sync between broadcast and broadband media content in an HbbTV scenario. The MediaSync module needs to perform the initial and continuous sync among the media streams. Once clock skew is corrected, the media streams are multiplexed in a single MP2T stream to be sent to the media player. In [87], the initial-sync process in the MediaSync module is presented (See Fig. 33). It uses the information from RTP/RTCP protocols [3] within the broadband stream and the MPEG-2 PSI/DVB SI tables [61] in order to initialize the initial sync. The assessment is performed, synchronizing an MP2T stream with an audio MP3 file. The continuous sync between the video and the audio is performed by detecting the clock skew in the audio and video streams. Then, the clock skew correction is performed in the MP3 audio stream before multiplexing into the final MP2T stream.

In [88], a system to synchronize Real-time subtitles with the audio/video streams at the source side is designed. The system generates a new media stream, from a broadcast TV channel, with embedded subtitles with the previously corrected timestamps, which is then delivered via an IPTV channel. The main objective of this technique is to eliminate the few seconds delay in live subtitling due to the subtitle generation process. The process consists of the creation and timestamping of the subtitles in real-time, at the moment the speech takes place, based on time references within the audio stream. This differs from other approaches, where the inter-relationships between subtitles and audio/video are established at packetization time, previously to the transmission or broadcast over the network. Therefore, subtitles and video/audio are out of sync. Once the timestamps in subtitles are properly corrected, a new MP2T is created, including the embedded original subtitles, synchronized to the audio/video streams. The proposed system

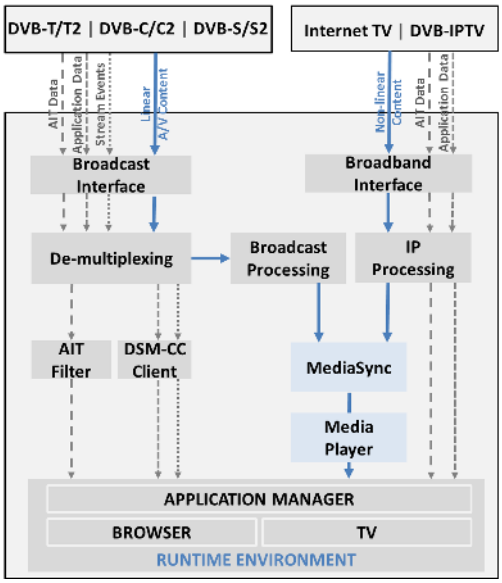


Fig. 33: HbbTV functional components with MediaSync module included [86]

solves the problem of the lack of sync in live subtitling, providing users a good QoE. The original TV channel is delivered via broadcast. Meanwhile, a few seconds later, the operator delivers the same stream with synchronized live subtitles via an IPTV channel.

The works in [89] and [90] describe the benefits of combining the two main mass media delivery systems, broadcast and broadband, at the receiver-side. A user's terminal architecture is designed to synchronize two MP2Ts, one delivered via broadcast and another via broadband technologies. The main objective in [89] is to improve the QoE of TV viewers and to '*free broadcast resources*' [89] by using Dynamic Broadcast systems. The proposed solution aims to exploit the two delivery systems by dynamically combining/interchanging MP2T streams from hybrid systems. The method implies the delivery of TV channels with a big audience via broadcast, while TV channels with small numbers of viewers could be delivered via broadband. This takes advantage of the two-way communication system provided by the broadband technology because a user's terminal can send back the audience feedback

TABLE XV: Summary references group of concepts (Some references could apply to multiple categories)

Standard	References to standards	References to implementations (or proof-of-concepts)
Media Delivery		[1], [2], [29], [30]
Media Sync	[12], [15], [16]	[5], [7], [10], [11], [13], [14], [17], [18], [19], [22], [23], [24], [25], [26], [27], [28], [64], [65], [86], [87]
MPEG-1	[38]	
MPEG-2	[21], [39]	[72], [73], [81]
MPEG-4/ MPEG-4 ISO	[44], [45], [46]	[40], [41], [42], [43], [73], [82]
HAS/ MPEG-DASH	[4], [31], [32], [33], [47], [48]	[49], [50], [51], [52], [66], [88], [89], [90], [91], [97]
MMT	[9]	[6], [53], [54], [55], [56], [57]
DVB	[35], [60], [61], [62], [71]	[37], [58], [59], [63], [92], [99]
HbbTV/Hybrid Sync	[8], [34], [70]	[36], [67], [68], [69], [93], [94], [95], [96], [98]
RTP/RTCP	[3], [75], [76], [77], [78], [79], [80], [83]	[20], [74], [81], [82], [84], [85]

about a specific program to the system. The TV delivery system, after analysing the information, can react and decide which delivery method to use for delivering the TV channel. Media sync is performed for the same MP2T stream delivered via both delivery technologies, broadcast and broadband, to improve the performance of the hybrid TV system. It suggests delivering the same MP2T via broadband and broadcast and when the system decides between one or the other, then it can swap MP2Ts using buffering techniques, as well as time aligning timestamps (PTS, DTS) and clock references (PCR), to sync both streams and provide a seamless switching process to the user.

This functionality of the Dynamic Broadcast System is extended in [90] with time-shifted control delivery. Based on the user's preferences, the TV system decides to pre-store a TV program at the receiver-side for a future play-out. Therefore, the delivery time (i.e., arrival time) differs from the play-out time (i.e., the presentation time).

In [64], two additional media sync related scenarios are exploited. On the one hand, media sync is performed between media content sent via a broadcast FM audio stream and via a broadband MP2T stream. This is achieved by using a shared UTC clock and by inserting clock references within the Radio Data System (RDS) structures in the FM stream and within the TDT (Time and Date Table) inserted in the MP2T stream. On the other hand, media sync is also performed between a broadcast delivered MP2T stream and a broadband delivered MP2T stream. In such a case, media sync is achieved by precisely inserting timestamps within the TDT tables in both streams. That work argues that media sync could also have been achieved when using MPEG-DASH (by using the NTP timestamps carried within *ProducerReferenceTime* boxes in MPEG-DASH Segments) and when using RTP (by using NTP timestamps provided by RTCP packets).

Media sync is also used to enhance Rich Media Languages, such as HTML5, used with adaptive HTTP-based streaming. The work in [91] uses MPEG-DASH and describes Rich Media services conveyed within an adaptive HTTP-based media session along with the video and data, while ensuring tight sync. This feature is especially indicated for real-time interactive (web-based) services.

A traditional mechanism, used both in broadcast and broadband, is the usage of clock references and timestamps (i.e., PCR, DTS, PTS) fields within MP2T streams as a temporal reference [92] [93]. However, this mechanism has some limitations: first, the clock references can be overwritten by different components in the distribution chain, causing discontinuities in the clock references that affect the sync process; and, second, it is only valid if the different sources come from the same media provider.

In [94], a mechanism presenting the above limitations is proposed. Moreover, it does not follow the standard rules for the broadband delivery of RTP data. Other alternatives to accomplish the hybrid sync consist of using additional media streams, or mapped information such as watermarks or fingerprints [65]. However, the drawbacks are: low precision, higher overload, noise sensibility and poor scalability.

In [95], a solution is proposed to unify the broadband and broadcast technologies via the use of IP networks and the insertion of common temporal marks to achieve hybrid sync. Moreover, in [64], another solution is presented, based on the use of global clocks, carried in-band in IP networks, that neither requires the feedback channel nor implies any dependence relation in broadband and broadcast networks. However, it does not achieve a very high level of sync accuracy.

The works in [36], [65], [66], [67] and [69], use the ETSI 102 823 proposal [35], which consists of adding absolute temporal references for specific content/events (absolute content/event timelines), as auxiliary packets multiplexed in the broadcast stream. These temporal marks are not affected by PCR discontinuities, and therefore accomplish precise sync levels. One advantage of such a solution is that it is independent of the content type. However, this only solves part of the problem. Additional mechanisms are addressed to improve precision and interactivity as well as to provide appropriate signalling information.

Furthermore, analogous mechanisms are needed in the broadband streams to facilitate hybrid sync. In [65], event timelines are inserted in RTP streams to accomplish hybrid sync. In [66], event timelines are introduced in MPEG-DASH streams, with control information about these streams in the broadcast streams. In EU-FP7 HBB-NEXT[21] project, hybrid

[21] http://www.hbb-next.eu

sync solutions for HbbTV have been proposed, contributing to its standardization. They are based on the previously described mechanism of inserting event timelines into the broadband and broadcast streams. The Romeo (Remote Collaborative Real-Time Multimedia Experience over the future of Internet) project has also targeted the media sync area as part of their solution, although its main focus is on the delivery of 3D media to homes and mobile devices.

In the same way, to accomplish IDES, in [65], [67] and [69], simple mechanisms are defined to establish the connection and interchange of useful information (signalling and timing) among the involved devices.

In [96] media sync is used to provide high quality 3D multiview video via HbbTV in an environment combining DVB and peer-to-peer (P2P) technologies. The basic 3D service, the stereoscopic view, is delivered via DVB-T, whereas the other views are delivered via broadband (P2P). This guarantees that all users receive the basic 3D service and users with broadband access have the added views to enrich the 3D main view. Media sync is achieved by adding the PCR values in the chunk headers (a chunk conveys multiple MP2T video packets) and using PTS/DTS values since MP2T packets are identical in both, the DVB and the P2P streams.

The media sync solution proposed in [97] includes a *DASH-Time* PES (dedicated PES which conveys 64-bit *presentation_time* and *activation_countdown* fields and 8-bit *mimeType* and *URL_location* fields) in the broadcast stream to link the MP2T stream to an MPD MPEG-DASH location. The *DASHTime* PES also conveys a *presentation_time* associated to the PTS values in the broadcast stream.

The solution provided in [98] synchronizes a main (broadcast) and secondary (broadband) media stream, e.g., a broadcast movie stream with subtitles located at a supplementary server. Three steps are taken to achieve the sync goal, first calculating the synchronization delay, second finding in the supplementary client the packet for initial synchronization (based on the *start_time_offset* which is the '*time difference between original start time and the actual start time of the supplemental stream*' [98]) and '*the propagation delay between the supplemental server and client devices along with the processing time by the supplemental server*' [98] and finally, keep the flow of the new synchronized stream. After the initial synchronization, an Absolute Presentation Time (APS) is calculated every time a media packet is received from the supplementary server. The APS is the difference between the timestamp of the present packet, minus the timestamp of the first packet of the media stream in the supplementary server, divided by the supplementary server clock frequency.

Finally, broadband proof-of-concept of the TEMI solution (Section IV-B) is presented in [99]. The test bed uses a broadcasted MP2T video stream with TEMI timecode insertion. The video codec employed is HEVC and Scalable HEVC (SHVC), whereas the broadband delivery uses MPEG-DASH as a media delivery of ISO BMFF segments. A TEMI timecode is inserted for each video frame and represents the frame presentation time. GOP and segment duration are 2s and the size/duration of the play-out player is set to 3s. Three uses cases are studied in [99]: multi-layer video coding quality improvements (services offering spatial scalability), delivery enhancements through broadband (catch-up, fast rewind and fast forward-to-live scenario) and content personalization and accessibility (such as a sign language video stream for hearing-impaired viewers).

VII. Conclusion

The integration of broadcast and broadband delivery technologies for multimedia services enrichment is a reality. In this heterogeneous context, the sync of the play-out of different media content from different sources and through different delivery networks is a challenge. To accomplish media sync at the receiver-side in a media session, three areas have to be studied: how the encoder's clock system is reconstructed at receiver-side (via clock references and timestamps); how media streams play-out are synchronized at receiver-side; and, which standards and protocols are used to convey such information. Synchronization for multimedia can relate to either synchronizing from an absolute time, e.g., UTC, or timing/frequency. Depending on the context, either or both may be important. To reconstruct the sender's clock system at the receiver-side, media standards use clock references. Once both sender and receiver's clocks are synchronized, the media delivery protocol uses timestamps to align the play-out of the involved media streams throughout a media session.

The multiple connectivity capabilities of modern consumer devices (e.g., connected TVs, PCs, smartphones, etc) as well as the new patterns in media consumption, in which multi-screen settings are becoming commonplace, facilitates the deployment of multi-network or multi-protocol content delivery services. The study of how the timelines are implemented in the MPEG, RTP/RTCP and DVB standards gives us a general view of the existing and potential solutions to achieve media sync.

There are multiple technologies, protocols and standards for media delivery for both broadband and broadcast, each of them with its own characteristics, benefits and drawbacks. The combined and coordinated use of these technologies can deliver many benefits, as explained in this paper. Moreover, it can contribute to a wider reach and availability of multimedia services. However, this diversity of technologies and formats brings an increase in complexity as well as in compatibility and inter-operability problems between technologies.

This paper has focused on one of the key challenges regarding media sync: providing synchronized video and audio when either the same delivery technology (for one or multiple streams) or multiple technologies are used, regardless of whether it is broadband or broadcast. Likewise, we have focused on individual receivers, and examined intra-stream, inter-stream and inter-sender/multi-source sync. Furthermore, the relevance of media sync has been reviewed and some examples have been introduced.

To provide synchronized services in heterogeneous and hybrid systems, we can consider two main approaches. The first and more revolutionary one is to employ protocol-independent solutions which solve the problems of heterogeneity, incompatibility and inter-operability. However, it is not a short

term and easy solution, because it would imply changing the involved technologies. The second one is the design of mechanisms to identify and align temporal dependence between multiple streams and different technologies. This is a more realistic short term solution in today's multimedia delivery systems. Accordingly, it reflects the relevance of the contributions of this paper.

The differences in the timing models (MPEG-2, MPEG-4, ISO BMFF, MPEG-DASH and MMT standards), and the different delivery technologies (e.g., broadcast and broadband), have been widely described in this paper. The study of the use of timelines through the MPEG standards gives an overall view of all the solutions implemented in media sync up to date.

The objective of this paper has been to provide an in-depth knowledge of the technologies in use and an understanding of the format of the temporal references included in such technologies. We also have looked at how and where they are inserted in the media streams to provide synchronized services, not only using a single technology, but also when simultaneously using multiple technologies, such as broadcast and broadband, in a coordinated manner.

A summary of research dealing with media sync solutions in such environments has also been included. The HbbTV and MMT standards represent an important improvement in Hybrid media delivery. Moreover, several European projects (e.g., HBB-NEXT) have also contributed to the development of standards for hybrid media delivery and synchronization.

ACKNOWLEDGEMENT

This work has been funded, partially, by the "Fondo Europeo de Desarrollo Regional" (FEDER) and the Spanish Ministry of Economy and Competitiveness, under its R&D&i Support Program in project with ref TEC2013-45492-R.

The first author would like to thank her employer, Ericsson Ireland, for providing an environment conductive to pursuing research outside working hours.

REFERENCES

[1] J. Maisonneuve, M. Deschanel, J. Heiles, L. Wei , H. Lu, R. Sharpe and W. Yiyan "An Overview of IPTV Standards Development". Broadcasting, IEEE Transactions on, vol. 55, n. 2, pp. 315-328. June 2009.

[2] B. Li, Z. Wang, J. Liu and W, Zhu. "Two Decades of Internet Video Streaming: A Retrospective View". ACM Transactions on Multimedia Computing, Communications and Applications, vol. 9, n. 33, pp. 1-20. October 2013.

[3] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson. RFC 3550, RTP: A Transport Protocol for Real-time Applications. Internet Engineering Task Force, Standards Track. July 2003.

[4] ISO/IEC 23009-1: 2012. Information Technology. Dynamic Adaptive Streaming over HTTP (DASH). Part 1: Media Presentation Description and Segment Formats. April 2012.

[5] L. Beloqui Yuste and H. Melvin. "Enhanced IPTV Services through Time Synchronisation". 2010 IEEE 14th International Symposium on Consumer Electronics (ISCE). vol., n., pp. 1-6. Braunschweig. June 2010.

[6] S. Aoki, K. Otsuki and H. Hamada. "Effective Usage of MMT in Broadcasting Systems". Broadband Multimedia Systems and Broadcasting (BMSB), 2013 IEEE International Symposium on, vol., n., pp. 1-6. June 2013.

[7] L. Beloqui Yuste and H. Melvin. "Inter-media Synchronization for IPTV: A Case Study for VLC". Digital Technologies, Žilina, Slovakia. November 2009.

[8] ETSI TS 102 796 V1.2.1 (2012-11) Hybrid Broadcast Broadband TV. 2012.

[9] ISO/IEC 23008-1: 2014. Information Technology. High Efficiency Coding and Media Delivery in Heterogeneous Environments. Part 1: MPEG Media Transport (MMT). June 2014.

[10] F. Boronat, R. Mekuria, M. Montagud and P. Cesar. "Distributed Media Synchronization for Shared Video Watching: Issues, Challenges, and Examples". Social Media Retrieval, Springer Computer Communications and Networks Series, pp. 393-431. 2013.

[11] W. J. Kooij, H. Stokking, R. van Brandenburg and P-T. de Boer. "Playout Delay of TV Signals: Measurement System Design, Validation and Results". Proceedings of the 2014 ACM International Conference on Interactive Experiences for TV and Online Video. TVX'14, pp. 23-30. Newcastle (UK). June 2014.

[12] D. Mills, J. Martin, J. Burbank and W. Kasch. RFC 5905, Network Time Protocol Version 4: Protocol and Algorithms Specification. Internet Engineering Task Force, Standards Track. June 2010.

[13] F. Boronat, J. Lloret, M. Garcia. "Multimedia Group and Inter-stream Synchronization Techniques: A Comparative Study". Elsevier, Information Systems 34 (2009), vol. 34, n. 1, pp. 108-131. March 2009.

[14] J. Ridoux and D. Veitch. "Principles of Robust Timing over the Internet". Queue - Emulators, vol. 8, n. 4, pp. 30-43. April 2010.

[15] C. Demichelis and P. Chimero. RFC 3393, IP Packet Delay Variation Metric for IP Performance Metrics (IPPM). Internet Engineering Task Force, Standards Track. November 2002.

[16] V. Paxson, G. Almes, J. Mahdavi and M. Mathis. RFC 2330, Framework for IP Performance Metrics. Internet Task Force, Standards Track. May 1998.

[17] "Timers, Timer Resolution and Development of Efficient Code". Date of access: 12th August 2015. http://download.microsoft.com/download/3/0/2/3027D574-C433-412A-A8B6-5E0A75D5B237/Timer-Resolution.docx. June 2010.

[18] A. S. Tanenbaum and A. Woodhull. "The Minix book. Operating Systems. Design and Implementation". 3rd Edition. Pearson Prentice Hall. 2006.

[19] D. Tsafrir, Y. Etsion, D. G. Feitelson and S. Kirkpatrick. "System Noise, OS Clock Ticks, and Fine-grained Parallel Applications". In Proceedings of the 19th Annual International Conference on Supercomputing (ICS '05). ACM, New York, NY, USA, pp. 303-312. 2005.

[20] H. Melvin and L. Murphy. "An Integrated NTP/RTCP Skew Detection Method and Compensation for VoIP Applications". Multimedia and Expo, 2003. ICME'03. 2003 IEEE International Conference on, vol. 2, n., pp. 537-540. July 2003.

[21] ISO/IEC 13818-1. Information Technology. Generic Coding of Moving Pictures and Associated Audio: Systems Recommendation H.222 (2000E). International Standards Organization (ISO/IEC). December 2010.

[22] M. Montagud, F. Boronat, H. Stokking and R. Van Brandemburg. "Inter-destination Multimedia Synchronization: Schemes Use Cases and Standardization". Multimedia Systems, vol. 18, n. 6, pp. 459-482. November 2012.

[23] A. J. Mason and R. A. Salmon. "Factors Affecting Perception of Audio-Video Synchronization in Television". British Broadcasting Corporation. BBC R&D Publications. White Paper WHP176. January 2009.

[24] ITU-R BT.1359. International Telecommunication Union/ITU Radio Communication Sector Relative Timing of Sound and Vision for Broadcasting.

[25] R. Steinmetz. "Human Perception of Jitter and Media Synchronization". Selected Areas in Communications, IEEE Journal on, vol. 14, n. 1, pp. 61-72. January 1996.

[26] ATSC Implementation Subcommittee Finding. Doc. IS-191. "Relative Timing of Sound and Vision for Broadcast Operations". June 2003.

[27] ETSI TR 103 010 v1.1.1 (2007-03). Speech Processing, Transmission and Quality Aspects (STQ); Synchronization in IP Networks- Methods and User Perception.

[28] Y. Bang, J. Han, K. Lee, J. Yoon, J. Joung, S. Yang and J. Rhee. "Wireless Network Synchronization for Multichannel Multimedia Services". Advanced Communication Technology, 2009. ICAT 2009. 11th International Conference on, vol. 2, n., pp. 1073-1077. February 2009.

[29] A. C. Begen, T. Akgul and M. Baugher. "Watching Video over the Web: Part 1: Streaming Protocols", IEEE Internet Computing, vol. 15, n. 2, pp. 54-63. March-April 2011.

[30] A. C. Begen, T. Akgul and M. Baugher. "Watching Video over the Web: Part 2: Applications, Standardization, and Open Issues", IEEE Internet Computing, vol. 15, n. 3, pp. 59-63. May-June 2011.

[31] R. Pantos. Internet Engineering Task Force. Internet-draft. HTTP Live Streaming. Draft-pantos-http-live-streaming-16. April 2015.

[32] HTTP Dynamic Streaming. Date of access: 12 August 2015. http://www.adobe.com/ie/products/hds-dynamic-streaming.html

[33] MS-SSTR. Smooth Streaming Protocol v20150630. June 2015.
[34] HbbTV Specification Version 2.0. HbbTV Association. 1st August 2015.
[35] ETSI TS 102 823 v1.1.1 (2005-11). Technical Specification: Digital Video Broadcasting (DVB); Specification for the Carriage of Synchronized Auxiliary Data in DVB Transport Streams. November 2005.
[36] C. Köhnen, C. Köbel and N. Hellhund. "A DVB/IP Streaming Testbed for Hybrid Digital Media Content Synchronization". Consumer Electronics - Berlin (ICCE-Berlin), 2012 IEEE International Conference on, vol., n., pp. 136-140. September 2012.
[37] J. Whitaker. "DTV Handbook". 3rd edition. McGraw-Hill, New York. 2001.
[38] ISO/IEC 11172-1. Information Technology. Generic Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5Mbits/s. International Standards Organization (ISO/IEC). 1993.
[39] ISO/IEC 13818-2. Information Technology. Generic Coding of Moving Pictures and Associated Audio information: Video (1995E). International Standards Organization (ISO/IEC). International Standards Organization (ISO/IEC). 1995.
[40] O. Avaro, A. Eleftheriadis, C. Herpel, G. Rajan and L. Ward. "MPEG-4 Systems: Overview". Signal Processing: Image Communication, vol. 15, n. 14-15, pp. 281-298. January 2000.
[41] G. Franceschini. "The Delivery Layer in MPEG-4". Signal Processing: Image Communication, vol. 15, n. 4-5, pp. 347-363. January 2000.
[42] C. Herpel and A. Elefthriadis. "MPEG-4 Systems: Elementary Stream Management". Signal Processing: Image Communication, vol. 15, n. 4-5, pp. 299-320. January 2000.
[43] C. Herpel. "Elementary Stream Management in MPEG-4". IEEE Transactions on Circuits and Systems for Video Technology, vol. 9, n. 2, pp. 315-324. March 1999.
[44] ISO/IEC 14496-1. Information Technology. Generic Coding of Audio-Visual Objects. Part 1: Systems (2010E). International Standards Organization (ISO/IEC). June 2010.
[45] ISO/IEC 14496-12. Information Technology. Generic Coding of Audio-Visual Objects. Part 12: ISO Base Media File Format. International Standards Organization (ISO/IEC). October 2008.
[46] ISO/IEC 14772-1. Information Technology.Computer graphics and image processing. The Virtual Reality Modelling Language. Part 1: Functional Specification and UTF-8 Encoding. International Standards Organization (ISO/IEC). December 1997.
[47] A. Zambelli. IIS Smooth Streaming Technical Overview. Technical Report. Microsoft Corporation, March 2009.
[48] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach and T. Berners-Lee. RFC 2616, Hypertext Transfer Protocol – HTTP/1.1. Internet Engineering Task Force, Standards Track. June 1999.
[49] Multimedia Group of Telecom ParisTech. GPAC Group. Date of access: 12 August 2015. http://download.tsi.telecom-paristech.fr/gpac/DASH_CONFORMANCE/TelecomParisTech/
[50] XML Schema Part 2: Datatypes Second Edition. Date of access: 12 August 2015. http://www.w3.org/TR/xmlschema-2/#rf-defn
[51] J. Le Feuvre, J-M. Thiesse, M. Parmentier, M. Raulet, and C. Daguet. 2014. "Ultra High Definition HEVC DASH Data Set". In Proceedings of the 5th ACM Multimedia Systems Conference (MMSys '14), pp. 7-12. ACM, New York, NY, USA. 2014.
[52] A. Seema, L. Schwoebel, T. Shah, J. Morgan, M. Reisslein. "WVSNP-DASH: Name-Based Segmented Video Streaming". Broadcasting, IEEE Transactions on , vol., n. 99, pp.1. February 2015.
[53] S. Aoki, K. Otsuki and H. Hamada. "Effective Usage of MMT in Broadcasting Systems". Broadband Multimedia Systems and Broadcasting (BMSB), 2013 IEEE International Symposium on, vol., n., pp. 1-6. June 2013.
[54] L. Youngkwon, P. Kyungmo, L. Jin Young, S. Aoki and G. Fernando. "MMT: An Emerging MPEG Standard for Multimedia Delivery over the Internet". Multimedia, IEEE, vol. 20, n. 1, pp. 80-85. Jan-March 2013.
[55] Y. Lim. "MMT, New Alternative to MPEG-2 TS and RTP". 2013 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), vol., n., pp. 1-5. June 2013.
[56] S. Kwang-deok, J. Tae-jun, Y. Jeonglu, K. Chang Ki and H. Jinwoo. "A New Timing Model Design for MPEG Media Transport (MMT)". Broadband Multimedia Systems and Broadcasting (BMSB), 2012 IEEE International Symposium on, vol., n., pp. 1-5. June 2012.
[57] G. Fernando. "MMT: The Next-Generation Media Transport Standard". ZTE Communications. Vol. 10 n. 2, pp. 45-48. June 2012.
[58] U.H. Reimers. "DVB. The Family of International Standards for Digital Video Broadcasting". Proceedings of the IEEE, vol. 94, n. 1, pp. 173-182. January 2006.
[59] A. J. Stienstra. "Technologies for DVB Services on the Internet". Proceedings of the IEEE. vol. 94 n. 1, pp. 228-236. January 2006.
[60] ETSI TS 101 154 v1.11.1 (2012-11). Digital Video Broadcasting (DVB); Specification for the use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream. November 2012.
[61] ETSI EN 300 468 v1.14.1 (2014-01). Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems. January 2014.
[62] ETSI TR 101 211 v1.11.2 (2012-05). Digital Video Broadcasting (DVB); Guidelines on Implementation and Usage of Service Information (SI). May 2012.
[63] W. Fischer. "Digital Video and Audio Broadcasting Technology. A Practical Engineering Guide". Signals and Communication Technology. A practical Engineering Guide, 2nd edition Berlin: Springer. 2008.
[64] C. Concolato, S. Thomas, R. Bouqueau and J. Le Feuvre. "Synchronized Delivery of Multimedia Content over Uncoordinated Broadcast Broadband Networks". In Proceedings of the 3rd Multimedia Systems Conference (MMSys '12). ACM, New York, NY, USA, 227-232. 2012.
[65] C. Howson, E. Gautier, P. Gilberton, A. Laurent and Y. Legallais. "Second Screen TV Synchronization". IEEE International Conference on Consumer Electronics, vol., n., pp. 361-365. Berlin. September 2011
[66] J. Le Feuvre and C. Concolato. "Hybrid Broadcast Services Using MPEG DASH". Proceedings of the Media Synchronization Workshop 2013. Nantes (France). October 2013.
[67] R. Brandenburg and A. Veenhuizen. "Immersive Second-screen Experiences Using Hybrid Media Synchronization". Proceedings of the Media Synchronization Workshop 2013. Nantes (France). October 2013.
[68] A. Veenhuizen and R. van Brandenburg. "Frame Accurate Media Synchronization of Heterogeneous Media Sources in an HBB Context". Proceedings of the Media Synchronization Workshop 2012. Berlin (Germany). October 2012.
[69] C. Köhnen, N. Hellhund, J. Renz and J. Müller. "Inter-Device and Inter-Media Synchronization in HBB-NEXT". Proceedings of the Media Synchronization Workshop 2013. Nantes (France). October 2013.
[70] HBB-Next, Deliverable D.4.3.1, Evaluation: Intermediate Middleware Software Components for Content Synchronization. May 2013. www.hbb-next.ue/documents/HBB-NEXT_D4.3.1.pdf
[71] Information Technology Generic Coding of Moving Pictures and Associated Audio Information. Amendment 1: Delivery of timeline for external data. Recommendation ITU-T H.222.0 (2014) Amendment 1. April 2015.
[72] G. Goldberg. "IPTV-ID-0087. RTP/UDP/MPEG-2 TS as a means of transmission for IPTV Streams". International Telecommunication Union (ITU). Telecommunication Standardization Sector, Study Period 2005-2008. Source: Cisco System Inc. USA.
[73] A. MacAulay, B. Felts, Y. Fisher. IP Streaming of MPEG-4: Native RTP vs MPEG-2 Transport Stream. White paper. Envivio. October 2005.
[74] F. Boronat, J.C.G. Guerri Cebollada and J. L. Mauri. "An RTP/RTCP Based Approach for Multimedia Group and Inter-stream Synchronization". Multimedia Tools and Applications. vol. 40, n. 2, pp. 285-319. November 2008.
[75] Y. Kikuchi, T. Nomura, S. Fukunaga, Y. Matsui and H. Kimata. RFC 3016, RTP Payload Format for MPEG-4 Audio/Visual Streams. Internet Engineering Task Force, Standards Track. November 2000.
[76] D. Hoffman, G. Fernando, V. Goyal and M. Civanlar. RFC 2250, RTP Payload Format for MPEG-1/MPEG-2 Video. Internet Engineering Task Force, Standards Track. January 1998.
[77] J. van der Meer, D. Mackie, V. Swaminathan, D. Singer and P. Gentric. RFC3640, RTP Payload for Transport of MPEG-4 Elementary Streams. Internet Engineering Task Force, Standards Track. November 2003.
[78] C. Zhu. RFC 2190, RTP Payload Format for H.263 Video Streams. Internet Engineering Task Force, Standards Track. September 1997.
[79] Y. -K. Wang, R. Even, T. Kristensen and R. Jesup. RFC 6184, RTP Payload Format for H.264 Video. Internet Engineering Task Force, Standards Track. May 2011.
[80] S. Wenger, Y. -K. Wang, T. Schierl and A. Eleftheriadis. RFC 6190, RTP Payload for Scalable Video Coding. Internet Engineering Task Force, Standards Track. May 2011.
[81] A. Basso, G. L. Cash, M. R. Civanlar. "Real-Time MPEG-2 Delivery Based on RTP: Implementation issues". Signal Processing: Image Communication, vol. 15, n. 1-2, pp. 165-178. September 1999.
[82] A. Basso, S. Varakliotis. "Transport of MPEG-4 over IP/RTP". Transactions on Emerging Telecommunications Technologies, vol. 12, n. 3, pp. 247-255. June 2001.
[83] R. van Brandenburg, H. Stokking, O. van Deventer, F. Boronat, M. Montagud and K. Gross. RFC7272, Inter-destination Media Synchronization (IDMS) using the RTP Control Protocol (RTCP). Internet Engineering Task Force, Standards Track. June 2014.

[84] M. Montagud and F. Boronat. "RTP/RTCP and Media Sync: A Review and Discussion of Future Work". Proceedings of the Media Synchronization Workshop 2013. Nantes (France). October 2013.
[85] M. Montagud, F. Boronat, H. Stokking and P. Cesar. "Design, Development and Assessment of Control Schemes for IDMS in a Standardized RTCP-based Solution", vol. 70, n., pp. 240-259. September 2014.
[86] L. Beloqui Yuste and H. Melvin. "Enhanced IPTV Services through Time Synchronization". The 14th IEEE International Symposium on Consumer Electronics (ISCE) Braunschweig. June 2010.
[87] L. Beloqui Yuste and H. Melvin. "Interactive Multi-source Media Synchronisation for HbbTV. Proceedings of the Media Synchronization Workshop. Berlin (Germany). October 2012.
[88] M. de Castro, D. Carrero, L. Puente and B. Ruiz. "Real-Time Subtitles Synchronization in Live Television Programs". Broadband Multimedia Systems and Broadcasting (BMSB), 2011, IEEE 6th International Symposium on. vol., n., pp. 1-6. June 2011.
[89] P. Neumann, J. Qi and V. Reimers. "Seamless Delivery Network Switching in Dynamic Broadcast Terminal Aspects". 2011 IEEE International Symposium on Broadcast Multimedia Systems and Broadcasting. June 2011.
[90] P. Neumann and U. Reimers. "Live and Time-shifted Content Delivery for Dynamic Broadcast: Terminal Aspects". Consumer Electronics, IEEE Transactions on, vol. 58, n. 1, pp. 53-59. February 2012.
[91] C. Concolato, J. Le Feuvre and R. Bouqueau. "Usages of DASH for Rich Media Services". In Proceedings of the 2nd Annual ACM Conference on Multimedia Systems (MMSys '11). vol., n., pp. 265-270. New York, USA. 2011.
[92] K. Matsumura, M. J. Evans, Y. Shishikui and A. McParland. "Personalization of Broadcast Programs using Synchronized Internet Content". Consumer Electronics (ICCE), 2010 Digest of Technical Papers International Conference on, pp. 145-146. January 2010.
[93] M. Armstrong, J. Barret and M. Evans. "Enabling and Enriching Broadcast Services by Combining IP and Broadcast Delivery". British Broadcasting Corporation. BBC R&D Publications. White Paper WHP 185, September 2010.
[94] U. Rauschenbach, W. Putz. P. Wolf and R. Mies and G. Stoll. "A Scalable Interactive TV Service Supporting Synchronised Delivery over Broadcast and Broadband Networks". International Broadcasting Convention, IBC Conference. September 2004.
[95] S. Aoki, K. Aoki, H. Hamada and Y. Kanatsugu. "A New Transport Scheme for Hybrid Delivery of Content over Broadcast and Broadband". 2011 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), pp. 1-6. June 2011.
[96] A. Lykourgiotis, K. Birkos, T. Dagiuklas, E. Ekmekcioglu, S. Dogan, Y. Yildiz, I. Politis, G. O. Tanik, B. Demirtas, A. M. Kondoz and S. Kotsopoulos. "Hybrid Broadcast and Broadband Networks Convergence for Immersive TV Applications". Wireless Communications, IEEE, vol. 21, n. 3, pp. 62-69. June 2014.
[97] J. Le Feuvre and C. Concolato. "Hybrid Broadcast Services Using MPEG DASH". Proceedings of the Media Synchronization Workshop 2013. Nantes (France). October 2013.
[98] S. H. Kim, C. Lee, S. Kang, K. Seo and T. Jung. "Timing Control for Synchronizing Multimedia Streaming over Heterogeneous Networks". Advanced Communication Technology (ICACT), 2013 15th International Conference on, vol., n., pp. 260-263. January 2013.
[99] J. Le Feuvre, V. NGuyen, W. Hammidouche, P. Marchal and P. Dupain. "A Test Bed for Hybrid Broadcast Broadband Services". Proceedings of the Media Synchronization Workshop 2015 with ACM TVX 2015. Brussels (Belgium). June 2015.

Lourdes Beloqui Yuste Lourdes Beloqui Yuste is currently an Engineer at Ericsson, Ireland. Received her PhD in Information Technology for NUI Galway, Ireland, in September 2015. She received a Master in Software Design and Development in NUI Galway, Ireland, in 2009. She graduated from 'Universitat Rovira i Virgili', Tarragona (Spain).

Fernando Boronat Fernando Boronat (M93, SM11), received the M.E. and Ph.D. degrees in telecommunication engineering from the Universitat Politècnica de València (UPV), València, Spain, in 1994 and 2014, respectively. After working for several Spanish Telecommunication Companies, he moved back to the UPV in 1996. He is an Assistant Professor in the Communications Department at the Gandia Campus where he is the head of the Immersive Interactive Media R&D group. His main topics of interest are Communication Networks, Multimedia Systems, Multimedia Protocols and Synchronization. He is involved in several IPCs of national and international journals and conferences.
Webpage: http://personales.upv.es/fboronat/Research/index_investig_en.html

Mario Montagud Mario Montagud (M09) was born in Montitxelvo (Spain). He studied Telecommunications Engineering at UPV (Polytechnic University of València), in Spain, and obtained the PhD degree from the same university in March 2015. Since then, he is a 1-year ERCIM postdoc fellow at CWI (The National Research Institute for Mathematics and Computer Science in the Netherlands). His topics of interest include Computer Networks, Interactive and Immersive Media, Synchronization and QoE (Quality of Experience). Mario is (co-) author of over 30 scientific and teaching publications, and has contributed to standardization within the IETF (Internet Engineering Task Force). He is member of the Technical Committee of several international conferences, co-organizer of the international MediaSync Workshop series, and member of the Editorial Board of international journals. Webpage: https://sites.google.com/site/mamontor/

Hugh Melvin Dr. Hugh Melvin is based at the Discipline of Information Technology, NUI, Galway. His research scope extends to Internet Multimedia QoS & QoE, Timing Systems, Wireless Technologies, Cybersecurity for Critical Infrastructure, and Energy & Environmental Informatics. Prior to NUI, Galway, he worked for 9 years in Power Engineering. He serves as CoChair on the US NIST Cyber Physical Systems Academic Timing group www.cpspwg.org and is a member of the steering group for the Time Awareness in Applications, Computers, and Communication Systems interest group www.taaccs.org, and the Industry-based Timing conference International Telecommunications Synchronisation Forum.
Graduated with a B.E from UCD in 1988, an MBA from University of Limerick in 1997, a Research M.Sc from NUI Galway in 2000, and a PhD from University College Dublin in 2004.