HbbTV-compliant Platform for Hybrid Media Delivery and Synchronization on Single- and Multi-Device Scenarios

Fernando Boronat, *Senior, IEEE*, Dani Marfil, Mario Montagud, Javier Pastor

***Abstract*— The combination of broadcast and broadband (hybrid) technologies for delivering TV related media contents can bring fascinating opportunities. It is motivated by the large amount and diversity of media contents, together with the ubiquity and multiple connectivity capabilities of modern consumption devices. This paper presents an end-to-end platform for the preparation, delivery and synchronized consumption of related hybrid (broadcast/broadband) media contents on a single device and/or on multiple close-by devices (i.e., a multi-device scenario). It is compatible with the latest version of the Hybrid Broadcast Broadband TV (HbbTV) standard (version 2.0.1). Additionally, it provides adaptive and efficient solutions for key issues not specified in that standard, but that are necessary to successfully deploy hybrid and multi-device media services. Moreover, apart from MPEG-DASH and HTML5, which are the broadband technologies adopted by HbbTV, the platform also provides support for using HLS and RTP/RTCP broadband technologies. The presented platform can provide support for many hybrid media services. In this paper, in order to evaluate it, the use case of multi-device and multi-view TV service has been selected. The results of both objective and subjective assessments have been very satisfactory, in terms of performance (stability, smooth playout, delays and sync accuracy), usability of the platform, usefulness of its functionalities, and the awaken interest in these kinds of platforms.**

***Index Terms*— Broadband Multimedia, Broadcast Technology, Digital TV broadcasting, DVB, HbbTV, Hybrid Broadcast Networks, Interactive TV, MPEG-DASH, Multi-Screen, Multi-View, QoE, Smart TV, Synchronization**

I. Introduction

A huge variety of delivery technologies, consumption devices and media contents are at consumers' disposal nowadays. Regarding delivery, media can be distributed via broadcast and/or broadband technologies. On the one hand, broadcast technologies, such as Digital Video Broadcasting (DVB), can concurrently deliver the same media content to a large number of consumers. 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, broadband technologies (e.g. IP networks) can provide interactive, bi-directional and adaptive services, tailored to the resources and/or preferences of the customers. However, they typically provide poorer performance in terms of scalability, stability and latency than broadcast technologies. In this context, media can be delivered by using different forms of IP-based downloading and streaming techniques, of which the latter are becoming more popular.

Two main alternatives can be adopted in streaming services: managed and unmanaged [1]. On the one hand, managed services typically operate within (controlled) walled-garden environments (e.g., IPTV). They mainly rely on *push-based* streaming, by making use of Real-time Transport Protocol and its companion RTP Control Protocol (RTP/RTCP), standardized in RFC 3550 [2]. They are especially suited for delay-sensitive and interactive services. On the other hand, unmanaged services can operate worldwide, and mainly rely on *pull-based* HTTP-based Adaptive Streaming (HAS) solutions. Their main advantages are adaptability, scalability, reliability, ubiquity and cost efficiency. Different vendors and standardization bodies have specified their own HAS solution: HTTP Live Streaming[1] (HLS) by Apple [3]; Dynamic Adaptive Streaming over HTTP[2] by ISO/IEC and MPEG [4] (DASH hereafter); HTTP Dynamic Streaming[3] (HDS) by Adobe; and Microsoft Smooth Streaming[4] (MSS) by Microsoft. HAS solutions are under unceasing improvement and are being increasingly adopted for broadband media delivery. For example, DASH has been recently adopted by the Hybrid Broadcast Broadband TV -HbbTV- standard [5], and by many popular media services, such as Netflix and YouTube).

Regarding consumption devices, apart from the wide

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

Authors are with the Universitat Politècnica de Valencia, Immersive and Interactive Media R&D group, Communications Department, 46730, Grao de Gandia, Valencia, SPAIN (e-mails: {fboronat@dcom, mamontor@, damarre@dcom, fjpastor@dib}.upv.es).

[1] https://tools.ietf.org/html/draft-pantos-http-live-streaming-20 (last access, October 2017)
[2] http://mpeg.chiariglione.org/standards/mpeg-dash (last access, October 2017)
[3] http://www.adobe.com/products/hds-dynamic-streaming.html (last access, October 2017)
[4] https://www.iis.net/downloads/microsoft/smooth-streaming (last access, October 2017)

availability of connected TVs, the proliferation and massive usage of different kinds of companion or secondary devices are a reality. On the one hand, these companion devices are equipped with multiple connectivity capabilities (WiFi, 3G/4G/5G, FM…). This allows the concurrent consumption of different media contents via the same or different delivery technologies. On the other hand, these companion devices can have heterogeneous resources, performance and capabilities, mainly in terms of bandwidth (BW) availability, support of technologies and media types, media processing and resolution displays.

This heterogeneous media ecosystem may indeed lead to rivalry and incompatibility issues, but it also brings new fascinating research and development opportunities. The goal is to achieve a seamless convergence, coordination and inter-operability between the available delivery technologies, leveraging their strengths and complementary characteristics, in order to no longer conceive them as isolated worlds, but instead as a unique hybrid media ecosystem. In addition, a user-transparent interaction and coordination between the available connected devices for media consumption is desirable to extend the media consumption possibilities. By overcoming these challenges, a new wave of innovative and enriched services, and even new business models, can become a reality. This is particularly relevant to TV operators and other stakeholders (e.g., device manufacturers, content providers...), since the broadcast TV content can be augmented by live or on-demand media content delivered via broadband technologies to provide enriched media services.

Several relevant examples of possible hybrid TV media services[5], together with the related demands, preferences and expectations of Spanish consumers are provided in [6]. Some examples are: the provision of multi-device and multi-view (or even free viewpoint) TV; the concurrent consumption of various video streams (using either a Picture-in-Picture – PiP – or a mosaic view, and even on different devices) or switching between them; the provision of spatial, temporal and color scalability [7]; tiled streaming (i.e., ultra-high resolution video services in which different spatial areas of the same video are delivered in different streams) [8]; etc.

On the one hand, this enriched TV media consumption paradigm and new hybrid TV media services have a big impact for entertainment purposes. On the other hand, they can also bring social benefits, so that users can feel more integrated and immersed when consuming media. Some examples are native audio language selection, inclusion of videos with sign language and customized/adaptive presentation of subtitles or audio descriptions. These can be very useful and valuable, especially for elderly consumers and for consumers with specific disabilities.

It could be technically feasible to provide most of these possible augmented media services using exclusively broadcast technologies. Nevertheless, it must be taken into account that all of them require additional BW (scarce and expensive asset in the broadcast domain), and that not all the customers may want to pay for (or even -freely- use) them. Therefore, the use of broadband delivery technologies to provide these augmented services to the interested customers becomes a more efficient strategy. In addition, it provides flexibility for requesting the particular additional contents, based on the customers' interests, and it allows the provision of customized and adaptive media services tailored to the heterogeneous resources and capabilities of the involved devices, as well as to the customers' preferences and/or needs.

An evidence of the efforts towards this direction is the recent HbbTV standard [5]. It provides mechanisms for harmonizing the delivery and consumption of interactive broadcast and broadband TV-related contents through connected TVs and companion devices.

Moreover, hybrid TV media services have caught the attention of the scientific community and public organizations in recent times. Many research projects financed by the EU in the last years have been focused on the use of hybrid technologies to offer enhanced TV services, mainly based on the use of HbbTV standard. Some examples are: HBB-NEXT[6], HBB4ALL[7], TV-RING[8], MEDIASCAPE[9] or MPAT[10], among others. Many of the research studies referenced in this paper come from contributions achieved under the umbrella of these projects.

In particular, the latest release of HbbTV (2.0.1 version, July 2016) specifies functionalities and provides guidelines to achieve a synchronized consumption of related hybrid media contents, either on a single device – Main Screen (MS) – and/or on different devices in multi-device or multi-screen scenarios[11] – one MS plus one or various Companion Screens (CS) –. However, at the time of writing this paper, commercial platforms implementing this latest version of HbbTV are still unavailable. In addition, HbbTV 2.0.1 does not provide specific solutions to key remaining challenges that need to be addressed to successfully deploy these kinds of hybrid TV media services. In particular, such challenges are mainly the following ones:

1) Signaling mechanisms to discover, associate and describe the available related media contents.
2) Interaction, and coordination mechanisms between the available consumption devices.
3) Full-fledged adaptive synchronization (sync, hereafter) solutions (including protocols, algorithms and adjustment techniques) to accurately time-align the concurrent consumption of the same or different, but related, media contents. This aspect is specially challenging due to the

[5] The hybrid TV media services concept refers to those interactive media services that concurrently make use of broadcast and broadband technologies to consume TV-related media contents.

[6] Next-Generation Hybrid Broadcast Broadband, https://www.facebook.com/HBBNEXT/ (last access, October 2017)

[7] Hybrid Broadcast Broadband for All, http://cordis.europa.eu/project/rcn/191771_en.html (last access, October 2017)

[8] http://www.tvring.eu/ (last access, October 2017)

[9] http://mediascapeproject.eu/ (last access, October 2017)

[10] http://mpat.eu/ (last access, October 2017)

[11] While multi-device scenarios refer to scenarios in which several devices are involved regardless they include a screen or not (e.g., devices playing only audio), multi-screen scenarios refer to those scenarios in which several devices with their corresponding screens or displays are involved (e.g., multi-view scenarios).

delay variability when delivering one or multiple media contents, from either the same or different providers, via either the same or different technologies, and consuming them on a single device (e.g. a Smart TV) and/or on multiple heterogeneous devices (e.g., in a multi-screen scenario), as will be explained in Section II.B.

The contribution of this paper is twofold: first, some key concepts, components and technologies for hybrid media delivery and synchronized consumption are summarized; and second, an end-to-end platform for the preparation, delivery and synchronized consumption of related hybrid (broadcast/broadband) media contents, on a single device and/or on different (close-by) devices is presented. The platform is compatible with HbbTV 2.0.1 standard, implementing its key features, but it additionally provides adaptive and efficient solutions for the three previously mentioned remaining challenges. Moreover, apart from DASH and HTML5, which are the broadband technologies adopted by HbbTV, the platform also provides support for using HLS and RTP/RTCP broadband technologies. The main components and modules of the platform, as well as the involved devices, are described in this paper.

Although the presented platform can provide support for many of the previously mentioned hybrid media services, the use case of multi-device and multi-view TV has been selected, and implemented, for the evaluation of the platform. Such a use case allows the content provider to offer a director-controlled DVB stream about an event (e.g., a concert, a sports or an e-learning event) augmented with additional camera views of the same event delivered via broadband technologies (e.g., DASH, HLS or RTP). According to authors' previous study in [6], it is a very relevant and highly valued use case, with high commercial potential.

The results of both objective and subjective evaluations for that use case have been very satisfactory, in terms of performance (stability, smooth playout, delays and sync accuracy), usability of the platform, usefulness of its functionalities, and the awaken interest in these kinds of platforms. A link to demo videos is also provided to show the capabilities and proper performance of the platform.

The contributions of this paper are very timely and relevant, as the developed platform and its functionalities allow augmenting the traditional and passive lean-back TV watching experience with a more interactive, immersive, personalized and lean-forward TV watching experience, opening the door to a new wave of fascinating TV-related services.

The remainder of this paper is structured as follows. Section II provides some background to help understanding the paper and its contributions. Section III reviews the related works, mostly focusing on proof-of-concept implementations making use of the technologies and components described in Section II. In Section IV, the end-to-end HbbTV-compliant platform is presented. Section V provides some evaluation results, and, finally, Section VI presents our conclusions and suggests some ideas for future work.

II. Background

This section introduces some key concepts, components and technologies that are helpful to better understand this paper and its contributions.

A. Hybrid Media Delivery

In hybrid media delivery, *M* media streams can be generated by *S* sources, and delivered via *N* networks, using *T* delivery technologies, to *D* consumption devices. In the context of this work, media delivery is considered hybrid if $N>1$ and/or if $T>1$. The platform presented in this paper is valid from the simplest scenarios (e.g., where only S>1 or N>1) to the most complete and challenging ones (e.g., where $M>1$, $S>1$, $N>1$, $T>1$ and $D>1$).

Figure 1 shows a scenario in which M=S=T=D=N=n, i.e., with n sources generating n related contents being delivered using n streams through n heterogeneous (hybrid) networks to a single (dashed arrows) or to n consumption devices.

B. Delay Variability Problem

One of the main problems to tackle in hybrid media services is the existence of (high) delay variability. In such services, timing artefacts introduced by the agents in the end-to-end multimedia delivery chain will result in time lags between media streams and between destinations. Figure 2 shows the difference (i.e., asynchrony) between the end-to-end (or playout) delays of different streams generated at the same time (streams 1 and 2). It is mainly due to the existence of various factors, some of which can be related to the source and the delivery technology, to the delivery network, and/or to the consumption device's features and conditions. Capturing, encryption (if used), encoding, packetization, delivery (traffic load, trans-coding or format conversion -if needed-, fragmentation and re-assembly of packets, multicast or dynamic routing strategies, improper queuing policies…), processing, depacketization, decoding, buffering, rendering and presentation delays, and even packet losses, can seriously disturb the original media timing. The existence of such factors can result in different (and time variant) end-to-end delays when delivering media content from one or different media sources to one or multiple devices, using different delivery technologies. Likewise, it can also happen when the same content/stream is simultaneously delivered through different technologies (e.g., stream 1 delivered through technologies 1 and 3 in Figure 2).

Such a delay variability in hybrid scenarios needs to be compensated for, by adopting a combination of sync solutions. Such solutions should include specification techniques, insertion and interpretation of timelines, buffering policies, monitoring algorithms, and adjustment techniques, as well as clock sync approaches, as explained in the paper.

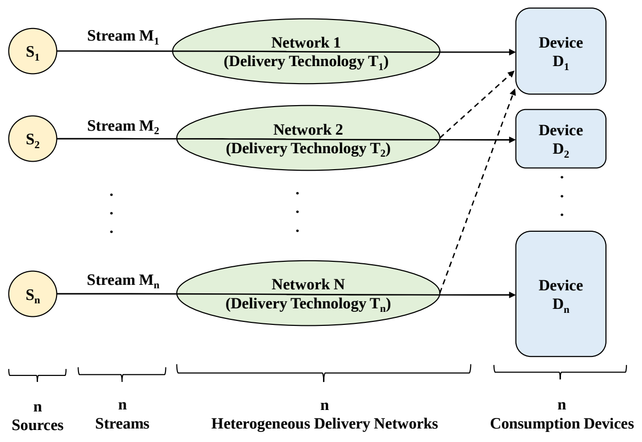

Stream M1
Network 1 (Delivery Technology T1)
Device D1
Stream M2
Network 2 (Delivery Technology T2)
Device D2
Stream Mn
Network N (Delivery Technology Tn)
Device Dn
S1
S2
Sn
n Sources
n Streams
n Heterogeneous Delivery Networks
n Consumption Devices

Fig. 1. Example of a hybrid media delivery scenario (M=S=T=D=N=n)

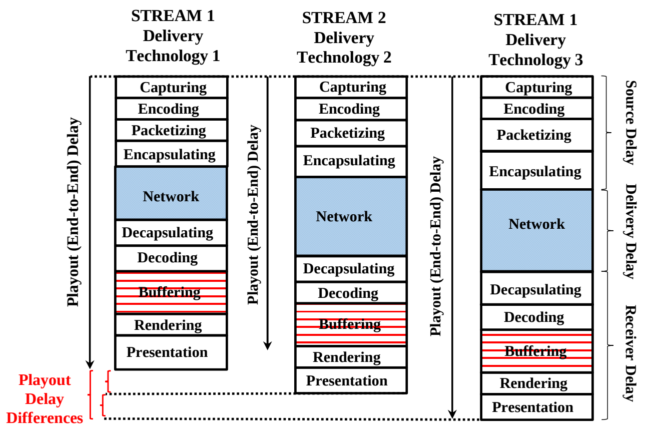

STREAM 1 Delivery Technology 1
STREAM 2 Delivery Technology 2
STREAM 1 Delivery Technology 3
Capturing
Encoding
Packetizing
Encapsulating
Network
Decapsulating
Decoding
Buffering
Rendering
Presentation
Playout (End-to-End) Delay
Source Delay
Delivery Delay
Receiver Delay
Playout Delay Differences

Fig. 2. End-to-end delay variability

C. Need for Hybrid Sync

To achieve a seamless convergence in hybrid media delivery, a key requirement is the concurrent sync of the media playout of the involved streams, which is referred to as *hybrid sync*, a particular sub-type of *inter-media sync* (a.k.a. *inter-stream sync* when the different media elements – audio, video…– are sent in different streams). By focusing on the content provider(s) side, the involved M media streams can be generated and/or sent by: 1) the same provider; 2) different, but somehow related, providers, which can agree on specific terms (e.g., on the use of a shared wall-clock source); or 3) completely independent providers, including the situation in which a third party performs trans-coding or trans-multiplexing to the original media content(s) along the delivery chain. By focusing on the consumer's side, if multiple consumption devices are involved (i.e., *D>1*), the term *Inter-Device Sync* (IDES) is commonly used when they are close-by (e.g., in a multi-screen scenario), while the term *Inter-Destination Media Sync* (IDMS) is commonly used when they are far apart. Whatever the particular case, it is very challenging to provide hybrid sync in this heterogeneous media ecosystem. First, the involved media streams can contain any kind of media type (e.g., audio, video, textual data, multi-sensory data…), with different formats and/or (processing and BW) requirements. Second, the involved hybrid (broadcast and broadcast) networks and delivery technologies can differ in terms of: 1) delivery nature (unicast, multicast or broadcast); 2) intrinsic delays; 3) provision of timelines; 4) availability of a feedback channel; etc. Third, the consumption devices can differ in terms of media processing capabilities and resources. In particular, the magnitudes of delays and delay variability are key issues when delivering media content(s), which are especially relevant in broadband delivery, mainly due to network and end-systems jitter. These delays are originated by different processes and steps along the end-to-end media delivery chain (see Figure 2) and can vary along different time instants and between different sources and consumption devices [9] [10] [11] [12]. Previous studies have reported on the magnitudes of these delays. First, in [9], it was pointed out that differences in end-to-end delays between receivers in an IPTV scenario can be larger than 6s. Second, the study in [11] showed that the delay differences when delivering the same media content via different broadcast variants and in different media formats can also accumulate up to 6s, and that HAS solutions can be more than 1 minute slower than broadcast technologies. In that work, it was also pointed that significant delay differences between receivers occur, even when the receivers use exactly the same TV delivery technology, setup combination (e.g., subscription type/quality) and equipment. However, no numbers were provided due to the lack of sufficient measurements from multiple geographically distributed sites.

All the above issues reflect the need for adaptive and accurate hybrid media sync solutions to compensate for the end-to-end delay variability in each of the mentioned situations. The next sub-sections introduce technological aspects and components that are very relevant to provide it.

D. Clock Sync

The availability of a coherent notion of time between the involved entities is a key requirement to achieve hybrid sync. On the one hand, it is necessary that the involved content providers or broadcasters insert common (or somehow related) timelines in order to time-align the consumption of the media streams delivered by each one of them. On the other hand, it is also necessary that the involved consumption devices have consistent timing information to accurately synchronize their playout processes. To accomplish this requirement, several alternatives for clock sync can be employed.

The most widely adopted solution is Network Time Protocol (NTP), specified in RFC 5905 [13]. It operates on a (hierarchical) client/server model and relies on the availability of an accessible NTP server to synchronize with. NTP is widely supported by different types of platforms and networking equipment. Moreover, many NTP servers are publicly available on the Internet. A drawback of NTP is that it relies on symmetric network delays, which is not a realistic assumption in current networks. However, sync levels in the order of a few milliseconds can be achieved by using NTP. Another alternative is to use Precision Time Protocol (PTP), specified in IEEE 1588-2008 [14]. PTP can typically achieve higher clock sync accuracy than NTP. However, PTP-aware elements and implementations are required, which may not be widely available yet. In this context, it is also noteworthy to remark that efforts are being devoted within the World Wide Web Consortium (W3C) Multi-Device Timing Community Group to provide a programming model for shared timing and motion in web-based multi-device scenarios, also relying on an external server. This approach has been proved to provide accurate results [15]. Moreover, it is also possible to adopt ad-hoc clock sync mechanisms, like the one implemented in GStreamer [16] or the one proposed in [8]. This can be an especially suited option for multi-screen scenarios, in which the delays between the involved devices are typically very low. In all these cases, the sync with a remote server might not be the most proper option, because the shorter the latencies, the smaller errors in the delay measurements and, therefore, higher clock sync accuracy can be achieved.

Additionally, more recently, the Wi-Fi Alliance[12] announced the introduction of Wi-Fi Certified TimeSync[13] for delivering precise time synchronization, within the sub-microsecond range, between multiple Wi-Fi devices. It could be a potential solution for IDES in the near future.

E. Timelines Insertion in MPEG2-TS for Hybrid Sync

The MPEG-2 Transport Stream (MPEG2-TS hereafter) format constitutes the backbone of the current broadcast media delivery ecosystem. MPEG2-TS specifies how audio, video and other data packets, from one or several TV channels, called *programs* according to the MPEG2-TS terminology, are conveyed within a continuous stream of bytes. In order to ensure proper media sync, time (i.e., clock references) and timing (i.e., timestamps) information is inserted within the streams. In particular, time information is provided via the Program Clock Reference (PCR) field, while timestamps are provided via the Decoding, Presentation and Composition Timestamp (DTS, PTS and CTS) fields, depending on the use of MPEG2 (DTS/PTS) or MPEG4 (DTS/CTS) encoding. These mechanisms provide *intrinsic* relative timelines for synchronizing the information within the same TS, but their value has no signification outside the media included in the TS. In order to synchronize two different MPEG2-TS, it is necessary that the involved providers utilize the same wall-clock source for inserting these relative timelines. Another important drawback of using PCR and timestamp field values for hybrid sync is that the temporal relationships can be overwritten by networking equipment (e.g., multiplexers, transcoders, splitters…) throughout the end-to-end delivery chain, without the content provider being aware of this transformation, which negatively affects the sync process.

To overcome such limitations, additional extensions to MPEG2-TS have been defined to provide *extrinsic* and absolute timeline information. First, the ETSI (European Telecommunications Standards Institute) TS 102 823 [16] specification defines a mechanism, called *ETSI Timeline* mechanism, to allow inserting timelines into the MPEG2-TS. It consists of inserting a broadcast timeline descriptor and a TV-Anytime (TV-A) descriptor into the payload of newly generated Timeline Packetized Elementary Stream (PES) packets (Figure 3). The new PES has to be advertised in the MPEG2-TS Program Map Table (PMT) as private data (*stream_type = 0x06*, *stream_id=0xBD*). For each I-frame, a PES packet with a timeline descriptor is generated, including the PTS value from the I-frame's PES header together with the *broadcast timeline descriptor* carrying absolute time values. After generation, the Timeline PES packets are multiplexed into the MPEG2-TS.

Since both the Video and Timeline PES contain the same PTS values, accurate hybrid sync can be achieved, regardless of PCR discontinuities. It is because the absolute timelines are inserted in the TS by the content provider and they will not be altered by any elements throughout the end-to-end delivery chain. Accordingly, the *ETSI Timeline* mechanism allows achieving hybrid sync even when different content providers are involved. Moreover, these timelines are content and transport agnostic, so they can also be used for the sync of specific events, arbitrary data and on-demand media streams that do not usually provide sync-related information.

More recently, a more flexible and BW efficient[14] extension to MPEG2-TS than the *ETSI Timeline* mechanism has been proposed by MPEG and DVB, as an amendment to ISO/IEC 13818-1, under the name of TEMI (*Timing and External Media Information*) [19]. TEMI mainly provides the following features:

- Insertion of extrinsic, absolute and stable timelines within the MPEG2-TS.
- Insertion of URLs (indicating the location of the related broadband contents augmenting the broadcast services) within the MPEG2-TS.
- Announcement of when the additional media contents will become active, by sending countdown signals for a given timeline identifier.

The first two types of metadata are commonly known as *TEMI timeline* and *location* (or *URL*) descriptors, respectively. These descriptors can be inserted (with a configurable insertion rate) into the MPEG2-TS in two manners (see Figure 4): 1) in the Adaptation Field (AF) included in the header of the MPEG2-TS transport packets (i.e., the TS header), as also shown in Figure 5; or 2) in a dedicated PES (which has to be advertised in the PMT as private data, with *stream_type = 0x26* and *stream_id=0xBD*, i.e. private stream). The first option is more efficient in terms of BW usage.

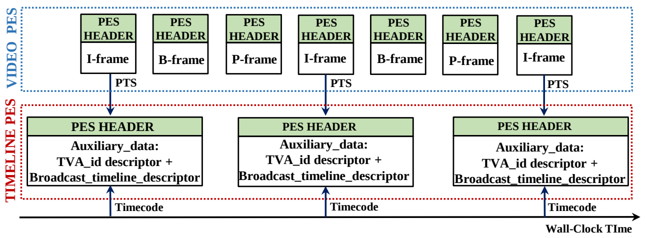


Fig. 3. Timeline insertion mechanism specified in ETSI TS 102 823 [16]

[12] https://www.wi-fi.org/who-we-are (last access, October 2017)

[13] https://www.wi-fi.org/discover-wi-fi/wi-fi-timesync (last access, October 2017)

[14] According to [18], for a 50Hz video stream, the timeline insertion overhead is reduced from 75 Kbps when using the ETSI Timeline solution [17] to 4 Kbps when using TEMI [19].

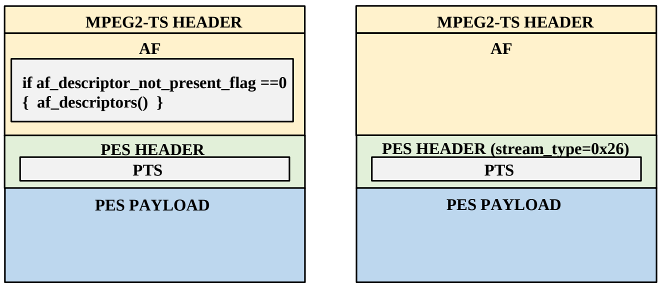


Fig. 4. Two options for TEMI insertion

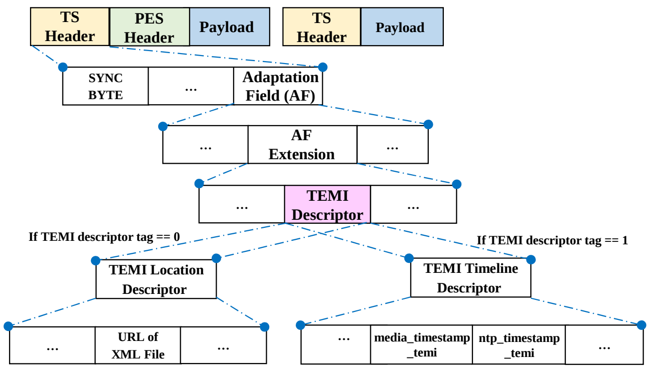


Fig. 5. Insertion of TEMI descriptors in the AF

F. Bootstrapping and Content Matching

In order to provide hybrid sync on a single device and/or on multiple devices, it is necessary to provide proper mechanisms for bootstrapping and content matching. On the one hand, bootstrapping refers to the process in which the involved consumption devices retrieve and launch a specific application that will help identifying and receiving the different available media streams, which we call *hybrid delivery app*. On the other hand, content matching refers to the process in which the temporal, spatial and/or semantic dependences between the available streams are retrieved for a proper and synchronized media consumption experience.

In conventional media streaming services, bootstrapping information can be provided by using either out-of-band or in-band approaches. An example of the former is the use of Session Description Protocol (SDP [20]) when using RTP/RTCP, while an example of the latter is the PMT when using MPEG2-TS. For hybrid delivery, the use of the PMT is not satisfactory, because an association between media streams from different sources is necessary. An alternative is to signalize the existence (and, optionally, the scheduling) of the related contents via broadband, by means of, e.g., SDP, sharing an Electronic Program Guide (EPG) or any other proprietary mechanism.

Two approaches could be followed to provide the bootstrapping information: server-based or stream-based. In the server-based case, the *hybrid delivery app* queries a central server (e.g., hosted by the broadcaster) about the location and metadata of the additional available media contents. Then, the server will send the responses with the requested information via the aforementioned methods (SDP, EPG…). In the stream-based case, this information could be embedded into the MPEG2-TS (e.g., by means of descriptors containing URLs of metadata files with the required information). In the presented platform, the latter approach is adopted.

G. Device and Service Discovery

Other essential aspects for hybrid sync are the discovery of secondary playout processes (running either on the same hybrid terminal or on companion devices), and notifying them of the existence of related contents to be played out.

Therefore, initial service and/or device (in multi-device scenarios) discovery processes should be launched to discover or associate different internal playout processes (single device scenario), and/or different playout processes running on different close-by devices (multi-device scenario).

In multi-device scenarios, in which IDES is required, another mechanism is also needed for the association between those devices involved in the session (only for these ones, as there could be other non-involved devices in the same network). This mechanism will allow the involved devices to be aware of the existence of other devices, and create a shared synchronized session, establishing a communication channel between them to exchange the necessary information for a proper interaction and sync.

Regarding this device discovery process, existing solutions requiring any active input by users could be used (e.g. pairing, typing passcodes or PIN codes, or scanning QR codes). However, it is desirable to use more user-friendly, transparent and lightweight solutions to maximize cross-platform support and the chances of successful deployment.

Regarding service discovery, existing solutions allow obtaining available content information (before establishing a -shared- sync session) in an automatic and transparent way for the end-consumer.

Solutions like UPnP (Universal Plug and Play) and DLNA (Digital Living Network Alliance) technologies could be candidates for these purposes. However, they are not widely and uniformly adopted in existing platforms (which may arise inter-operability problems) and their implementation would require significant efforts. A more appropriate alternative may be to utilize a combination of Multicast DNS (mDNS) and DNS Service Discovery (DNS-SD), together also known as ZeroConf or Bonjour[15], as proposed in [8]. It is a simple, lightweight, but powerful method for service and device discovery, which is widely implemented in the major platforms and adopted by the industry [8].

More recently (after the publication of [8]), the DIscover And Launch (DIAL) protocol [21] has been specified as a simple mechanism that allows companion devices (CS) to discover and launch applications on main terminals (MS). DIAL enables a seamless integration of CS with the media consumption experience on the MS. It relies on UPnP, Simple Service Discovery Protocol (SSDP), and HTTP protocols. It follows a client/server approach, and is composed of two parts:

1. *DIAL Service Discovery*, which enables a DIAL client to discover DIAL servers on its local home network and

[15] https://support.apple.com/bonjour (last access, October 2017)

obtain access to the DIAL REST Service on those devices;

2. *DIAL REST Service*, which enables a DIAL client to query, launch and, optionally, stop applications on a DIAL Server.

DIAL is maintained by Netflix and YouTube, with inputs from a variety of partners, and has been also adopted by HbbTV standard. Accordingly, the presented platform relies on DIAL for providing device and service discovery.

H.DVB-CSS: Digital Video Broadcasting - Companion Screens and Streams

Direct communication between the Smart TV (MS) and companion devices (CS) is expected to provide the most reliable, responsive and accurate mean of providing IDES. The DVB group[16], composed of TV manufacturers, broadcasters, and platform operators, has contributed to the specification of both use-cases and commercial requirements for companion screen experiences (via the DVB CM-COS Group[17]) and participated in subsequent standardization activities, leading to the *DVB Companion Screens and Streams* specification (DVB-CSS) [22] [23]. Concretely, DVB-CSS defines a standardized and interoperable framework that allows applications on smart companion devices (CS) to construct an integrated, synchronized content experience with the main TV (MS) over the same home network. DVB-CSS includes protocols that enable a TV to indicate to companion devices ‘what content it is presenting’ and ‘the current timeline of the piece of content being presented’, thereby supporting reliable, accurate, and timely interactions over the home network [24]. DVB-CSS allows the development of compelling and interoperable implementations of CS products. It considers the use of TEMI [19], in combination with WebSocket, UPnP and SSDP protocols [18] [24]. In particular, it is composed of different interfaces, building blocks and protocols to achieve different purposes [18]:

1. Device Discovery and Association.
2. Content identification and exchange of state information. It implies the usage of the SSDP protocol for discovering the MS.
3. WallClock (WC) Sync. The WC sync protocol (called CSS-WC) provides the means to use a clock sync algorithm to establish a best effort approximation of the MS’s WC at the CS application.
4. Timeline Synchronization (TS). Exchange of timestamp information between the involved (close-by) devices. The CSS-TS protocol is used by the MS to report about time positions on its timeline (timelines that are signaled in the broadcast or derived from the streaming container format).
5. Events triggering (e.g., detection and triggering of *“Do It Now events”*[18] in the broadcast stream).

[16] https://www.dvb.org/about (last access, October 2017)

[17] https://www.dvb.org/groups/CM-COS (last access, October 2017)

[18] DVB based application platforms, like MHP (Multimedia Home Platform) and HbbTV, use ‘*Do It Now stream events*’ which do not have a defined trigger time. Instead, they should be triggered immediately after being received.

I. HbbTV Standard

HbbTV is a global initiative aimed at harmonizing the broadcast and broadband delivery of entertainment services to consumers through connected TVs, set-top boxes and companion devices. HbbTV 2.0.1 [5] is the latest version of the standard, released in July 2016. Compared to the earlier versions of the standard, it specified new functionalities and features to achieve hybrid sync on a single device (i.e., hybrid terminal) and/or on multiple devices (i.e., between MS and CSs). Some of its most relevant features (within the context of this work) are:

- It makes use of conventional DVB-T (and, therefore MPEG2-TS) services for the delivery of broadcast contents. Although some extensions to the DVB streams are added, the HbbTV specification ensures backward compatibility for those DVB receivers not supporting it.
- Adoption of contemporary web technologies, such as HTML5, CSS (Cascading Style Sheets), JavaScript and WebSockets.
- Adoption of DASH as the broadband technology to deliver additional related media contents.
- Adoption of DIAL for device and service discovery. The discovery process is underpinned by the use of the SSDP protocol to locate DIAL servers on the local network segment and obtain access to a DIAL REST service on those devices. Endpoint locations for app-to-app communication (explained later) and IDES are also discovered by issuing a query to this interface.
- It proposes mechanisms for launching a CS application from an HbbTV application on the MS, and vice versa, i.e., remotely launching an HbbTV (MS) application from a CS. DIAL is used for these purposes. The DIAL REST service enables a DIAL client (on a CS) to query, launch, and optionally stop MS applications on a DIAL server device.
- Application to Application (App-to-App) Communication Service. This service uses a WebSockets server (WSS) to facilitate direct communication between HbbTV-related apps running on MS and CS. Hereafter, *App2App WSS* is used to refer to this server.
- HbbTV proposes the use of a so-called Application Information Table (AIT), included in the DVB stream, for signaling the availability of HbbTV applications. The content provider or broadcaster uses the AIT to include URLs pointing to those applications that should be launched upon switching to a particular TV channel.
- (Hybrid) Sync between related media streams or between media streams and applications on single hybrid terminals. TEMI solution is adopted for these purposes.
- (Hybrid) Sync between related media streams or between media streams and applications across devices in a multi-screen scenario. For this purpose, HbbTV has adopted the aforementioned DVB-CSS specification.

The presented platform adheres to HbbTV 2.0.1 regarding most of the previously mentioned features and technologies, except for some of the DVB CSS parts (e.g., event triggering), which inclusion is left for future work.

Likewise, the launching of CS Apps can be achieved by implementing the DIAL server functionality on companion devices and client functionality on the main device. Nevertheless, this will raise security and battery lifetime issues on companion devices. Furthermore, it requires companion devices to run services in the background.

III. Related Work

This section mainly focuses on reviewing the most relevant works within the context of hybrid sync, most of them relying on the solutions introduced in the previous section, and the developed platforms including hybrid sync solutions. The most relevant proof-of-concept implementations for IDES (within the context of this work) are briefly described as well. Media sync solutions relying on proprietary techniques, such as watermarking and fingerprinting, are not considered, due to their multiple drawbacks, as discussed in [18], and because they are not appropriate techniques within the context of this work. Examples of those drawbacks [18] are the low precision, high overload, poor interactivity, poor scalability, only work when the audio can be successfully detected –i.e. noise sensibility-, require access to proprietary backend infrastructure for audio analysis, etc.

A. *Overviews of Media Synchronization*

Over the years, many media sync solutions have been proposed for a variety of delivery technologies, networked environments and applications. Likewise, various studies have deeply analyzed the advances on media sync. The works in [10] and [25] provide a taxonomy and classification of existing inter-media sync, IDES and IDMS solutions, while the study in [26] provides a historical review of media sync, by also conveying the background of technological advancements, sync modeling and human perception issues. More recently, the work in [18] provides an overview of many available standards for inter-media sync, IDES and IDMS, including HbbTV. In addition, the work in [12] reviews how clock references (timing) and timestamps (time) are conveyed within different MPEG (including MPEG2-TS and DASH) and DVB standards.

B. *Media Synchronization solutions involving one single consumption device*

In [27], an algorithm to achieve inter-media sync between MPEG2-TS streams generated by the same provider is proposed. It is based on controlling the audio/video playout processes according to the values of the PCR and associated PTS fields. It performs an *initial sync process* to ensure that the playout of the involved streams begins at the same time, by delaying the (slave) stream with greater PTS values until it is in sync with the other (master) stream, if necessary.

AT IBC (International Broadcast Convention) 2004[19], a prototype of a system developed in the SAVANT FP5 EU project[20] was presented. It provides end-to-end support to present two media streams at the receiver side in a synchronized manner, even if they are transmitted via different networks. On the one hand, at the content provider side, the timestamps for the main DVB content consist of NPT (Normal Play Time) descriptors inserted into the MPEG2-TS. The generation of the NPT descriptors and RTP timestamps (for broadband content) was triggered to start simultaneously. At the start of the main program, their values in both streams are reset and increased periodically and monotonically. This way, a common time base between the multiple hybrid media sources is stablished. On the other hand, at the client side, a local clock is generated based on (and synchronized to) the extracted NPT values. A limitation of that approach is that it assumes co-location of the different media sources. In addition, DVB regards the use of NPT to be obsolete and RFC 3550 [2] specifies that the initial value of timestamps must be randomly generated. Therefore, the current applicability of the proposed system/method is limited.

In [28], an approach to personalize broadcasted services using the Internet is presented. In it, the broadcaster provides additional contents via the Internet for service personalization, which are synchronized with the broadcasted content. The broadcast and Internet contents are transmitted at the same time (TCP is considered for transport in broadband). Accordingly, the receiver has to combine and synchronize both types of contents to generate an enjoyable personalized presentation. As a primary and simple implementation, a prototype of multi-language subtitles is presented. For sync handling, the system inherits the mechanism of MPEG-2 Systems used in conventional broadcasting systems. The content delivered via the Internet is marked with the PTS according to the same reference clock as in the broadcasting stream. Then, the receiver uses this reference clock as its internal clock, as in MPEG-2 based systems. A buffer is used to delay the presentation timing of the broadcast content, in order to compensate for the higher delay experienced by the broadband delivered content. The use of PCR and PTS values in MPEG2-TS to synchronize hybrid media streams is also analyzed in [29]. That work focuses on the use case of an alternative soundtrack using RTP-based streaming over an IP-based (broadband) delivery network to provide intelligibility for consumers who have difficulties in understanding speech when presented with background sound. However, in that case, the streams are generated by different sources, which shared a common wall clock source (e.g., by using the same NTP server) to insert PTS values into the media streams in a synchronized manner. A demonstration prototype is presented, but only considering a hybrid terminal, without companion devices.

In [30] and [31], the advantages and disadvantages of using MPEG2-TS over DVB, MPEG2-TS over IP, MPEG2-TS over RTP/IP and RTP/IP protocols for broadcast and/or broadband delivery are discussed (in [30], from the viewpoint of hybrid

[19] https://www.yumpu.com/en/document/view/30348279/demonstration-at-ibc-2004-the-school-of-engineering-and-design/7 (last access, October 2017)

[20] Synchronized and scalable AV content Across NeTworks, http://cordis.europa.eu/project/rcn/63328_en.html (last access, October 2017)

media sync). In [30], a solution for hybrid delivery and sync on broadcast and broadband networks, called Advanced Transport Scheme (ATS), is proposed. In it, the media components can be separately transported or multiplexed (i.e. transported together). Any media components on broadcast or on broadband networks can be flexibly combined to compose hybrid content. The media components (media data) and control metadata (control data) necessary for receiving ATS packets are separately transported. Consumers can obtain the contents in different ways: they can obtain content on a broadcast channel by means of its control metadata obtained on broadband networks; they can obtain content on a broadcast channel by obtaining its control metadata on the broadcast channel, as in conventional broadcast systems; and they can obtain content on broadband networks by obtaining its control metadata on broadband networks as well. Therefore, users can consume the available contents without being aware of the delivery channels/networks in use. The performance of the hybrid delivery system was tested on a prototype composed of the following components: content sender, broadcast (DVB-S) modulator and demodulator, network emulator, and content (hybrid) receiver. Nevertheless, the sync accuracy was neither evaluated nor discussed, and just statistics about the overhead for media and control metadata packets were provided.

In [31] and [32], the usage of MMT (MPEG Media Transport) standard [33] in hybrid delivery systems is explained. MMT is out of the scope of this paper, as the presented platform is compliant with the HbbTV specification and not with the MMT one.

In [34] and [35], the topic of hybrid media delivery is also investigated. In [34], a solution for hybrid sync between an audiovisual stream delivered via a broadband IPTV channel, using RTP/RTCP protocols, and an audio stream broadcast delivered via DVB (both streams in MPEG2-TS format) is proposed. No implementation was provided in that paper. In [35], the proposal is focused on the sync between an audiovisual MPEG2-TS stream delivered via an IPTV channel, and an MP3 audio stream delivered via Internet Radio. Both streams provide information about the same football match, which allows the customers the selection of their favorite commentator, and/or of a commentator in their native language. The proposed solution consists of removing the audio content from the main MPEG2-TS and adding the audio content from the secondary stream into the former one. Although the presented implementation is based on the use of RTP/RTCP protocols as the delivery technology for both streams, it is stated that the technical solution is also applicable when the main MPEG2-TS is delivered via DVB. A prototype was developed in Java, including the creation of two media streamers, one for the MPEG2-TS audiovisual stream and another for the MP3 audio stream. The sync goal in both works [34] and [35] is achieved by ensuring that all the involved content providers use a common wall-clock source (e.g., by using NTP), and then performing two main processes: *initial* and *continuous sync*. The *initial sync process* is based on using NTP-based timestamps for initially aligning the playout of the media streams. After *initial sync*, the *continuous sync process* is started to periodically correct clock skews between the different media streams, if exist. For RTP/RTCP-based delivery, the NTP-based timestamps from RTCP Sender Reports (SR) can be used [2]. For DVB-based delivery, the use DVB Program Specific Information (PSI) tables, such as the Time and Date Table (TDT), the Time Offset Table (TOT), and some modifications to the Event Information Table (EIT) is proposed. Since the DVB system does not use RTCP, another method is needed to include timing information. It is proposed to send the relationship between the (relative) MPEG2-TS timestamps from the related program with the (absolute) wall-clock time using the EIT. The EIT contains, among other information, the transport stream ID, event ID, start time and duration. In particular, it is proposed to add a 33-bit extra field indicating the timestamp of the initial PTS event called *PTS_timestamp*, as in the MPEG2-TS.

In [36], another solution for hybrid sync on single devices is proposed, which is also based on the use of global clocks and that does not require communication across networks, keeping them independent. In that work, two different scenarios are investigated, implemented and tested in a platform that is based on GPAC framework [40]. On the one hand, media sync is performed between an audio FM stream delivered via broadcast and an audiovisual MPEG2-TS delivered via broadband. It is achieved by inserting wall-clock timestamps within a Radio Data System (RDS) structure in the FM stream and within the TDT in the MPEG2-TS. The bootstrapping information is provided within the RDS structure of the FM channel, which involves a quite long latency (up to 1 min or even higher) for receiving the information about the existence and location of the related broadband streams. On the other hand, media sync is also performed between a broadcast MPEG2-TS and a broadband MPEG2-TS (using DASH). The bootstrapping information in this case relies on the DASH MPD file. In such a case, wall-clock timestamps are inserted within the TDTs of both streams. In that work, it is shown that the time differences between the values carried in TDT and PCR fields, together with clock drifts, might end up with a sync error higher than 1.1s. It is also observed that the time values in the TDT and PCR field can vary up to 2s, which can lead to a sync error in the order of 4s when using two MPEG-2 TS.

In [37] a sync mechanism for media contents from hybrid sources, like DVB and DASH, is proposed. It is based on the use of absolute broadcast timelines, starting from zero at the start of a show. For broadcast, the DVB broadcast timeline descriptor defined in the aforementioned *ETSI Timeline* specification [16] is used. That absolute timeline is inserted by the DVB broadcaster/DVB media encoder. For broadband, since all the segments in MPEG DASH have the exact same duration and an absolute starting time, the playout time of MPEG DASH content on the client side can be easily adapted to the absolute timeline. The segments' *start* codes are used as synchronization points. The *absolute ticks* field of the *broadcast timeline descriptor* is set to zero at the start of a TV show. Therefore, the start attribute specifies the start of the video relative to the TV show's start. Although the solution is targeted for hybrid media scenarios, the implemented testbed for evaluation is all-IP based (i.e., without broadcast DVB-T/S transmission).

In [38], it is demonstrated that accurate hybrid sync (in the order of few milliseconds) on a single device can be achieved, by combining the possibilities of utilizing the values of the PCR/PTS fields and also the *ETSI Timeline* mechanism [16] (as a common absolute clock reference inserted into each MPEG2-TS). This approach overcomes the issues of only relying on the PCR/PTS mechanism (e.g., as in [27] and in [29]). The media playout engine(s) can extract the embedded timelines, and adjust the media presentation times accordingly. In that work, a testbed based on the GStreamer framework [16], as the one presented in this work, is developed. In particular, the functionality of the MPEG2-TS de-multiplexer (demux) in GStreamer is extended to extract the embedded timelines and a custom GStreamer element is developed to compare timelines and adjust the playout processes accordingly. The evaluation is performed by playing out a remote MPEG DASH stream and local media files (simulating the additional broadcast content). The work in [39] presents a solution for hybrid delivery and sync on a single device, also based on the use of the *ETSI Timeline* mechanism and on the use of DASH for broadband delivery. It also proposes DASH-compatible tools to provide the location of the additional broadband content, as well as a description of the media contents being delivered. More recently, the same research group presented an evolved testbed for hybrid delivery and sync in [7], also based on the extension of the GPAC framework [40]. It provides support for: 1) Insertion and extraction of TEMI (Timeline and Location) descriptors [19] to achieve hybrid sync; 2) High Efficiency Video Coding (HEVC) and its scalable extension Scalable High efficiency Video Coding (SHVC) for the video encoding; and 3) DASH for the delivery of the broadband streams. In that work, the different SHVC encoded video layers are split in different tracks. The base layer track is sent via a broadcast channel (simulated with an IP multicast channel) and the enhancement layer is delivered via DASH. Then, the layered video is decoded and played out, combining the information received via the hybrid technologies.

C. Media Synchronization involving several consumption devices

The previous works have addressed the hybrid sync problem within single devices, but IDES is also becoming increasingly relevant due to the massive proliferation and usage of companion devices.

Regarding device and service discovery, different approaches are reviewed and compared in [41], from several point of views: a) the market solutions for them, b) the discovery and association challenges in distributed and pervasive computing, and c) the translation of the general scientific challenges to a TV centric home environment. Some solutions to be implemented over TVs or Set-Top-Boxes with HbbTV v1.1 or v1.5 are presented and validated in that work.

Regarding hybrid sync approaches, the work in [42] implements the *ETSI Timeline* mechanism [16], in combination with the PCR/PTS-based mechanism, to provide hybrid sync in second screen TV services. In that work, it is stated that the obtained sync accuracy is sufficient to provide lip-sync (i.e., inter-media sync between audio and video) between a multicast audio stream (via RTP) and a broadcast DVB-T video stream, but no results are provided. Similarly, the work in [8] proposes the combination of a set of technological components with the goal of providing immersive second screen experiences. In particular, that work includes the development of a tiled streaming solution for HLS (HAS), which allows users to freely navigate (e.g., pan/tilt/zoom) around an ultra-high resolution video panorama on their secondary screens, while watching the main director-controlled DVB stream on the main TV, in a synchronized manner. IDES is achieved by also using the *ETSI Timeline* mechanism [16] to associate the broadcast and broadband timelines, and by adopting an ad-hoc clock sync mechanism. Device discovery capabilities are also implemented, by using a combination of mDNS and DNS-SD, introduced in Section II.E.

In [43], an HbbTV framework for multi-screen scenarios is presented. It handles the discovery and connection mechanisms of HbbTV and second-screen devices' browsers plus the communication between applications running in those browsers. This framework also includes mechanisms allowing HbbTV applications to launch applications on secondary devices, as well as a communication channel enabling interaction between devices. However, sync mechanisms are neither mentioned nor described in that paper.

In [46] a new wake-up and sync mechanism for multi-screen applications using iBeacon[21] and Notification[22] technologies is presented. iBeacon uses a Bluetooth Low Energy (BLE) signal which can be detected by iOS devices. Any device supporting BLE can be turned into an iBeacon transmitter and alert applications on iOS devices nearby. Some ideas for a user-friendly remote launching mechanism of TV companion screen applications are proposed.

The works in [24] and [47] present prototypes with, as far as authors know, the first implementations of the different protocols and parts of the DVB-CSS specification [23], adopted by HbbTV, to enable a synchronized playout between a hybrid terminal and companion devices. In [24], a single pre-generated and locally stored MPEG2-TS file emulates the broadcast content. The following four use cases are presented: video to video sync; video to audio sync; video to CS web application sync; and responsive active and passive control of the TV depending on user action on a CS web application. Likewise, some benefits that can be provided by leveraging of the DVB-CSS functionalities are discussed. However, no objective and subjective assessments about the media sync performance are provided in that work. In [47] a TV emulator

[21] https://developer.apple.com/ibeacon/ (last access, October 2017)
[22] https://developer.apple.com/notifications/ (last access, October 2017)

based on an open source Python implementation[23] of the DVB-CSS protocols is used. The TV (MS) plays the main video content (e.g., a Shakespeare play in the tests). As in [24], the CS application also consists of a DVB-CSS client on an iOS device and has a web-based user interface. In this case, it plays the transcripts (subtitles) of the video contents on the MS with different forced asynchrony levels. Likewise, three different interaction modes (passive, exploration and call-to-action) are considered to assess their impact on the users' engagement and asynchrony noticeability. Results of a subjective assessment with 18 participants are presented, but no conclusive results about the delay/asynchrony perception thresholds are obtained. It is stated that for this type of companion screen experience, sync delays between -500ms and 1000ms are unlikely to be noticed. Moreover, it is also shown that users are significantly more distracted by the companion device's content (transcripts) when the asynchrony becomes higher.

In [44] a web-based distributed adaptation architecture for multi-device applications driven by media content is presented. It has been developed under the umbrella of the aforementioned MEDIASCAPE project. That architecture allows application developers create Web applications in terms of logic components once and add properties to define the multi-device application's behavior on a high abstraction level. The work also includes a deep analysis of the adaptation challenges for several multi-device applications use cases. Four adaptation challenges are identified, such as multi-device adaptation, user interface adaptation, personal and shared device adaptation and inter-components communication.

Recently, the work in [45], after analyzing the Hybrid TV ecosystem and the involved technologies, provides details on a large pilot deployment for a multi-device hybrid broadcast-broadband service for a live TV program. It defines and describes the development and deployment of an innovative live service that the Basque public broadcaster (Euskal Irrati Telebista) provided to cover the election night in the Basque Country, Spain, in September 2016. The use case included media streams of views from different cameras (HLS streams), social media, election results and statistics (graphics and tables). That paper also provides results regarding usage statistics of end users, and presents a discussion with some interesting lessons learned about the current hybrid broadcast-broadband ecosystem and multi-device live services. In that pilot, hybrid sync was not critical for the broadcaster, since the introduction of the necessary delays on the broadcast signal to enable it was not allowed. Instead, the broadcaster preferred to follow a best-effort approach of providing all the content when it was ready.

IV. HbbTV-compliant Platform for Hybrid Media Delivery and Synchronized Consumption

This Section presents the developed end-to-end platform for hybrid media delivery and sync. It is mainly comprised of two parts: the contents provider side and the consumer side. Figure 6 provides an overview of the platform, with both parts (framed into dashed boxes), together with the main involved modules and components (all of them explained in the following subsections). It allows the delivery of TV-related media contents via both broadcast (DVB-T) and broadband (DASH, HLS, and RTSP[24] + RTP/RTCP) technologies. The broadcast contents will be played out by an MS App (in the hybrid terminal) at the consumer side. In addition, the broadcast streams will include specific metadata to allow the involved CS App(s) to play out additional related broadband media contents, if desired. The CS App(s) can be integrated within the hybrid terminal and/or within companion devices (tablets, smartphones), as illustrated in Figure 6. Therefore, the platform is valid for both single- and multi-device scenarios, in which one or multiple MS and one or multiple CS can be involved.

The platform has been developed by mainly using the GStreamer framework [16], in both Linux- and Android-based devices. Nevertheless, as GStreamer provides cross-platform support, it can also be easily extended to provide support for iOS-based and Windows-based devices.

A. *Contents Provider Side*

At the contents provider side (left side of Figure 6), the platform includes different modules for the proper encoding, encapsulation, segmentation, storage, modulation and transmission of the media contents. Moreover, it also includes modules for the generation and insertion, or storage, of relevant metadata (e.g., absolute timelines, description and location of the available media contents…).

The platform allows the delivery of media contents via broadcast, using DVB-T, and via broadband, using DASH, which is the technology adopted by HbbTV for broadband delivery. Moreover, it allows the delivery of (broadband) media contents using HLS and traditional RTSP + RTP/RTCP streaming services. RTSP is specified in RFC 2326 [48], whereas RTP and RTCP are specified in RFC 3550 [2]. These protocols are especially suitable for low latency services over managed environments (e.g., IPTV).

1) *Broadcast*

Regarding broadcast delivery, the platform includes several modules and functionalities:

- Proper encoding of the media contents (e.g., resolution, frames per second or fps, bitrate, Group of Pictures or GoP patterns…) by using the *ffmpeg* framework[25].
- Their encapsulation in MPEG2-TS.
- Addition of the TEMI timeline and location descriptors [19], by using the MP42TS tool of the GPAC framework [40]. The rate of insertion of both TEMI descriptors can be configured, but the TEMI timelines are typically added once per I-frame. In a future work, a custom GStreamer element to insert the TEMI descriptors and AIT data into MPEG2-TS streams will be developed to have a fully Gstreamer-based platform.

[23] https://github.com/BBC/pydvbcss (last access, October 2017)

[24] Real Time Streaming Protocol, RFC 2326 [48]

[25] https://ffmpeg.org/ (last access, October 2017)

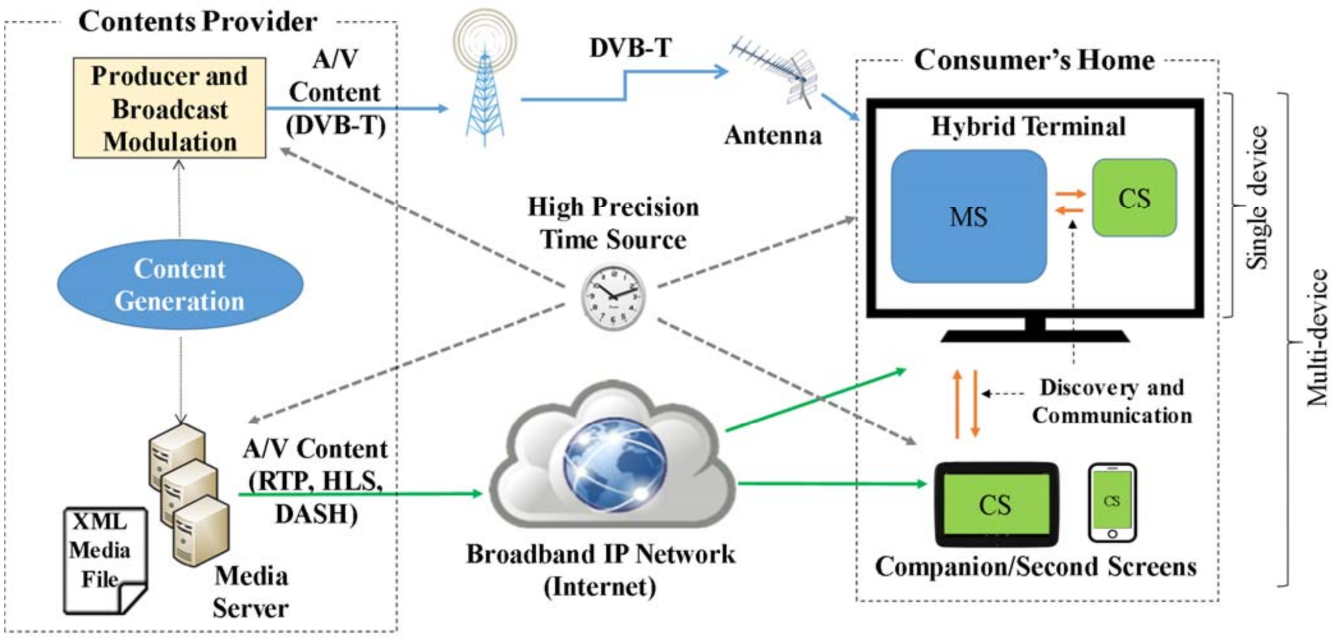


Fig. 6. Overview of the end-to-end HbbTV-compliant platform for hybrid media delivery and synchronized consumption

- DVB-T modulation and transmission, by using the Dektec DTA-2111 PCI (Peripheral Component Interconnect) card (hardware) and the DekTec StreamXpress software.

2) Broadband

On the one hand, regarding HAS (DASH and HLS), the platform includes different modules for: 1) multi-quality media encoding, using the *ffmpeg* framework; 2) media segmentation, with a configurable length of segments; 3) index file generation; and 4) storage of the (DASH and HLS) segments and index files on a conventional web server. More details about the DASH modules and the involved procedures can be found in [49]. In that work, our group presented an end-to-end platform for the adaptive delivery and playout of DASH contents. Similar modules and processes are used for HLS.

On the other hand, regarding RTP streaming, the contents can also be encoded in different qualities. An RTSP + RTP/RTCP streaming server has been developed, by making use of libraries and plugins natively provided by the GStreamer framework. RTP is used for media delivery, RTCP for the exchange of control messages, and RTSP for the media session control. It is possible to deliver the media contents in both multi-unicast and multicast modes, over UDP, but also to convey the RTP/RTCP streams and the RTSP messages over TCP, in a multiplexed manner. More details about the RTSP server and clients can be found in [50], in another previous work of our group.

3) Clock Sync

The platform makes use of a common wall-clock server (NTP server) to coherently insert timelines in all the involved (broadcast and broadband) media streams (see Figure 6). It will allow their time-aligned presentation at the consumer side, on single devices and/or on multiple devices.

4) Contents Signaling, Discovery & Description

Apart from the generation and preparation of all the related hybrid contents, as well as the insertion of the TEMI timeline descriptors, an additional mechanism needs to be included to announce the availability of the related broadband media contents and to provide the necessary metadata about them. These metadata include the relationship with the broadcast contents, content types, URLs for their access, etc. This way, receivers can be aware of the existence of such related contents and have all the required information to decide if they want to play them out, and, if so, prepare the proper media players.

As the media contents signaling, discovery and notification mechanisms, the TEMI location descriptor [19], with a configurable insertion rate in the MPEG2-TS, is used. In particular, it includes a URL of an XML file, which will be used as the hybrid media contents matching and description mechanism[26]. This XML file is stored on a web server (e.g., the same one used to store the media contents), and can be accessed via HTTP by the consumers' devices. This XML file can be updated during the media session lifetime (e.g., when broadband contents are dynamically generated).

Depending on the provided hybrid media service, the XML file can include different entries and attributes to link the available hybrid media contents and to provide relevant information about them. Such information can be the delivery technology (e.g., DASH, HLS, RTP…), the type of content (e.g., audio, video, web…), the encapsulation format (e.g., MP4, MPEG2-TS…), the encoding type or format (e.g., H264, MP3, HTML5…), and other relevant metadata (e.g., a brief

[26] According the ISO/IEC 13818-1:2013/DAM6 (Anex T.3.1), "The location descriptor is used to signal the location of external data that can be synchronized with the program".

description of the content, the relationships with the broadcast content, language…). In addition, it includes other relevant information, such as the last time it was updated, and the clock technology (e.g., NTP, PTP…) in use as the clock source (e.g., to insert timelines) at the contents provider side, together with the URL of the employed clock server and the absolute timestamp (according to that clock) at which the particular event started.

Table I describes and summarizes some of the tags and properties that can be used in this file. Figure 7 shows an example of the XML file for the multi-view scenario used in the evaluation section. Metadata for several available camera views (inside <*MEDIA*> tags, with different <source> elements in order to specify alternative origins for the same content) is included. Moreover, the file contains the last time it was updated (<*LASTUPDATE* > tag), and the global clock technology and server in use (<*CLOCK*> tag). A URL to related HTML5 contents (using the <*WEB*> tag) is also provided (e.g., to the official website of the event or program being broadcasted). Specific identifiers or links to Social Media platforms could also be added to allow the consumers getting more information about the broadcast program or event, so they can comment on it or discuss about it with other customers. A typical example could be the inclusion of a Twitter hashtag (e.g., by using, the <*TWITTER*> tag).

TABLE I. SUMMARIZED TAGS AND PROPERTIES USED IN THE XML FILE

TAG	Property	Description
MEDIA	specifies the necessary metadata for any available AV media content	
	id	unique identification value for the element
	media_type	(media) content type
	media_format	format or encoding information
	metadata	brief description
	temi_init	absolute global time when the content generation started
Source		allows to specify alternative origins for the same content
	protocol	used protocol
	uri	uniform resource identifier
WEB	specifies the necessary metadata for any available (related) data on web sites	
	id	unique identification value for the element
	protocol	used protocol
	media_type	(media) content type
	media_format	format or encoding information
	metadata	brief description
	uri	uniform resource identifier
LASTUPDATE		specifies the last time the file was updated
	media_format	format of the time value
	value	time and date value
CLOCK	Specifies the metadata for the global clock technology	
	id	unique identification value for the element
	protocol	used protocol
	media_type	(media) content type
	media_format	format of the time value
	metadata	brief description
	uri	uniform resource identifier

```
[…]
<Hybrid Media Contents File>
<MEDIA id="1" media_type="AV" media_format="h264/aac"
            metadata="front_camera/english"
            temi_init="3699255471291907022">
      <source protocol="http/dash"
            uri="http://192.16.0.10/multicam/dash/cam1.mpd"/>
 </MEDIA>
 <MEDIA id="2" media_type="AV" media_format="h264/aac"
            metadata="first_row_camera/english"
            temi_init="3699255471291907022">
      <source protocol="http/hls"
            uri="http://192.168.0.37/multicam/hls/cam2.m3u8"/>
 </MEDIA>
 <MEDIA id="3" media_type="AV" media_format="h264/aac"
            metadata="back_camera/english"
            temi_init="3699255471291907022">
      <source protocol="rtsp" uri="rtsp://224.0.0.10:5001/cam3"/>
 </MEDIA>
 ...
  <WEB id="1" protocol="http" media_type="website"
            media_format="html5"
            metadata= "url_event/english"
            uri="http://iim.webs.upv.es/multiview_app"/>
  <CLOCK id="1" protocol="ntp" media_type="time"
            media_format="64_bit_ntp_time"
            metadata="" uri="193.145.15.15"/>
  <LASTUPDATE protocol="http" media_type="time"
          metadata="" format="dd/mm/yyyy-hh:mm:ss"
          value="11/04/2017-11:20:00"/>…
</ Hybrid Media Contents File >

[…]
```

Fig. 7. Example of the XML File for a Multi-View Scenario

B. Consumer Side

At the consumer side (right side of Figure 6), the platform includes different modules and components. They are needed to allow the discovery, selection, reception, processing and adaptive playout of the available hybrid media contents in a synchronized and personalized manner, on hybrid terminals and, if desired, on companion devices. This sub-section describes all of them and their interactions.

1) Playout of Hybrid Media Contents

The involved consumption devices can include one or more hybrid terminals and none, one or several companion devices, all of them with the required HbbTV-compliant functionalities.

The hybrid terminals will integrate an MS Module, including a MS App, which will be responsible of receiving, tuning, processing and playing out the DVB-T (MPEG2-TS) contents. The GStreamer pipeline (i.e., chain of elements to perform a media-specific task) to create and configure the DVB-T player is shown in the upper part of Figure 8. A brief explanation of the functionalities of each element is provided in Table II. These elements are natively provided by the GStreamer framework. Nevertheless, the functionalities of the *tsdemux* element (inside a red dashed rectangle) have been extended in this work to be able to retrieve the TEMI descriptors from the incoming broadcast MPEG2-TS.

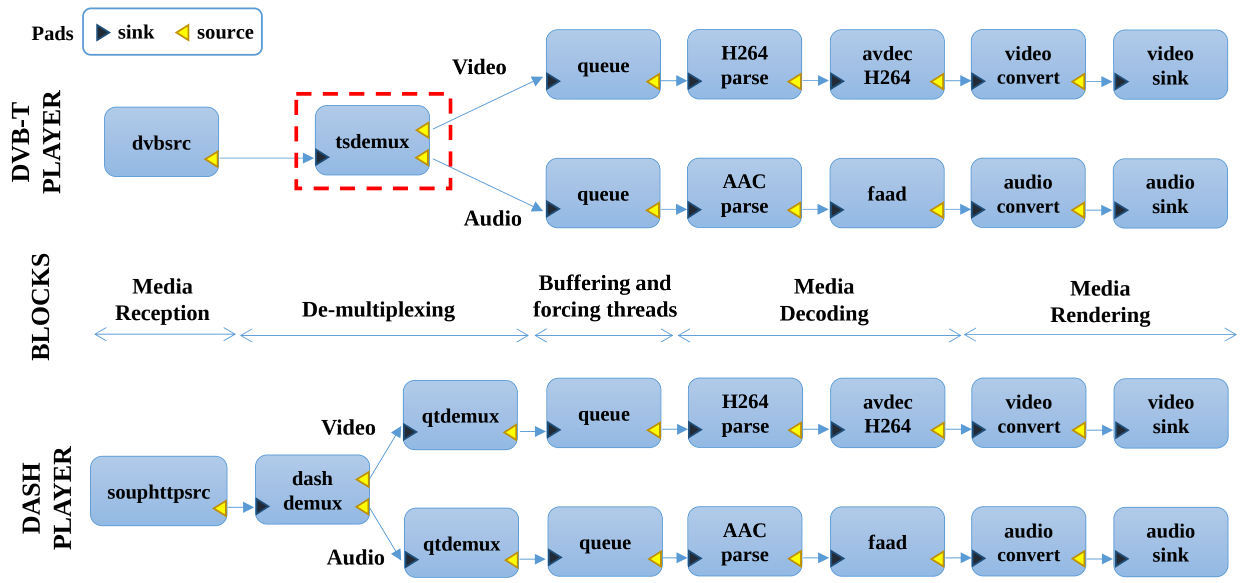


Fig. 8. Example of the Gstreamer Pipelines for the DVB-T and DASH Players

Apart from the broadcast contents, the hybrid media services can also include related broadband contents, enriching the consumption experience. These broadband contents can be played out by the same hybrid terminal and/or by companion devices. For these purposes, the involved entities will integrate CS Modules, including CS Apps, which will be responsible of receiving, processing and adaptively playing out the broadband (DASH, HLS and RTP) media contents, in a synchronized manner with the related hybrid contents.

The GStreamer pipeline to create and configure the DASH player is shown in the lower part of the Figure 8. A brief explanation of the functionalities of its elements is also provided in Table II.

The GStreamer pipelines to create the HLS and RTSP + RTP/RTCP players are not shown (because they are similar to the one for DASH). Nevertheless, the specific elements needed in them have also been listed in Table II.

The platform supports the simultaneous playout of hybrid broadcast and broadband contents, using different combinations for all the mentioned technologies.

It also allows the inclusion of related HTML5 media contents, but, in this case, a coarse-grained sync process with the related contents is applied, as achieving highly accurate sync at the stream-level in such a case is currently not possible.

The pipelines for broadband contents are dynamically created when the consumers indicate, via the Graphical User Interface (GUI), their willingness to consume any of the available broadband contents. In particular, the GUI includes a menu with the available contents, and some playout controls, such as VCR controls[27] (“play”, “pause” and “seek”), volume level adjustment, and playout configuration modes (e.g., PiP, mosaic view and full-screen).

The functionalities (including the MS and CS Modules) of the hybrid terminal have been developed in Linux-based devices, while the functionalities of the companion devices (only including the CS Module) have been developed both in Linux-based and Android-based devices.

2) Device and Service Discovery & App Launching

The platform requires proper functionalities to enable a user-transparent discovery, association and interaction between the involved MS and CS Modules, as well as for the corresponding App launching. These steps are essential to enable a synchronized consumption of hybrid media contents on a single device and/or on multiple devices (multi-screen scenarios).

As HbbTV 2.0.1 relies on DIAL and SSDP protocols to achieve these purposes, our platform includes modules with their required functionalities. On the one hand, an open-source implementation (using *Node.js*[28]) of the Companion Screen functionalities[29] included in HbbTV 2.0.1 has been adopted and extended to provide the functionalities of such protocols.

Figure 9 shows the components and the different communication functionalities involved in the platform. In particular, a DIAL server component has been integrated within the MS Module. It will be enabled upon launching that module and will be continuously waiting for requests from DIAL client components included in CS Modules. On the other hand, the functionalities provided by the SSDP protocol at the CS Module have been developed from scratch. The component implementing the SSDP protocol will be launched on the CS Module every time a discovery process of a DIAL-enabled hybrid terminal (MS Module) is required. After the discovery process, an association between the MS and CS Modules will occur, and a bi-directional channel between them will be available and used to launch HbbTV apps and

[27] The VCR controls should be disabled when hybrid sync needs to be provided, as their execution will originate a loss of sync between the involved player and the other active ones.

[28] https://nodejs.org/ (last access, October 2017)

[29] https://github.com/fraunhoferfokus/node-hbbtv (last access, October 2017)

exchange relevant information.

The MS Module has some elements that are intended to allow the establishment of a bidirectional communication channel between MS and CS Apps. As it can be seen, the MS Module needs to have running processes, such as the App2App WSS or the DIAL server. Moreover, the CS Module also needs to implement SSDP functionalities in order to discover any available MS Module. Regarding communication between MS and CS Modules, two different communication modes, multicast and unicast, (between both MS and CS Modules and Apps) are used in a sequential way, wherever the CS Module is executed (either in the same hybrid terminal or in a secondary device). First, multicast mode is used to enable the CS Module to discover any active MS Module, by sending a multicast SSDP *M-SEARCH* message and receiving the corresponding unicast response(s) (continuous red lines in Fig. 9). The response from the MS Module includes enough information to let the CS Module get the URL of the App2App WSS component of the MS Module. This component is the one that allows forwarding messages in unicast mode, between MS and CS Apps, through a bidirectional WebSocket-based channel (discontinuous green line in Fig. 9), during all the session.

3) Discovery of the Related Hybrid Media Contents

As described in Section IV.B.1, the developed DVB-T player (in particular, the Gstreamer *tsdemux* element of its pipeline) of the hybrid terminal has been modified to be capable of retrieving the TEMI descriptors [19] from the incoming MPEG2-TS. The TEMI location descriptor will give access to an XML file providing the necessary information for a proper playout of the related hybrid media contents. That file will be requested (via HTTP), and parsed by the MS App, which will send the information of interest to the involved CS Apps, through the stablished bidirectional channel.

With the information of the XML file, a menu with a list of the available contents (including metadata about them) will be dynamically created. Accordingly, users can check them and decide if they want to consume them, on the hybrid terminal and/or on the companion devices.

Figure 10 presents a flow diagram of the different processes executed in the MS Module to provide the described functionalities (and additional ones) and their interactions. Firstly, the MS Module launches the DIAL server, which allows to be identified by any available CS Module. Afterwards, the MS Module synchronizes its own clock with a global (NTP) clock reference. Then, the MS App checks if there is any available broadcast stream. If so, the player starts playing out the broadcast content while extracting TEMI data. Simultaneously, the MS Module listens to new CS Modules' requests to join the session (through the App2App communication channel). After these steps are done (extraction of TEMI data and listening to new requests for connection), the MS App will send them the TEMI data (initial data to new CS connections and timing data to already connected CS App(s)) through that channel in order to enable a synchronized multi-screen scenario. This process will loop until the end of the media session.

TABLE II. GSTREAMER ELEMENTS USED TO IMPLEMENT THE BROADCAST AND BROADBAND MEDIA PLAYERS' PIPELINES

Element (content)		Description
dvbsrc (DVB-T)		Receives, tunes and demodulates the DVB-T signal from a DVB card
udpsrc (UDP)		Receives and processes UDP packets from the IP network
souphttpsrc (HTTP)		Receives and processes data from a remote location specified by a URI, via HTTP/HTTPS
rtpmp2tdepay (RTP)		Extracts the MPEG2-TS from RTP packets (RFC 2250)
tsdemux		It processes the MPEG2-TS, identifying and splitting (i.e., demultiplexing) the available programs and streams (audio, video, …)
dashdemux (DASH)		DASH demuxer. Analyzes the MPD and requests, in each iteration, the segment of the most appropriate quality (by default, according to the available BW)
hlsdemux (HLS)		HLS demuxer. Similar to the previously described *dashdemux* element
qtdemux		Demuxer for MP4 (QuickTime) contents, separating the video (H264) and audio (Advanced Audio Coding, AAC) for their decoding
queue		Commonly used as a data queue or buffer, but it can also be used to force independent execution threads for each of the outputs of a demuxer (e.g., tsdemux) element for a better performance. The second functionality is the reason why it is used in the presented platform
Audio Decoder Block	*aacparse*	AAC audio stream parser
	faad	Free MPEG-2/4 AAC decoder
	audioconvert	Audio format converter (if needed)
Video Decoder Block	*h264parse*	H.264 stream parser
	avdec_h264	H264 decoder
	videoconvert	Video format converter (if needed)
autovideosink		Automatically selects the most appropriate video sink (i.e., output), according to the device in use, its hardware and Operating System (O.S.)
autoaudiosink		Automatically selects the most appropriate audio sink (i.e., output), according to the device in use, its hardware and Operating System (O.S.)

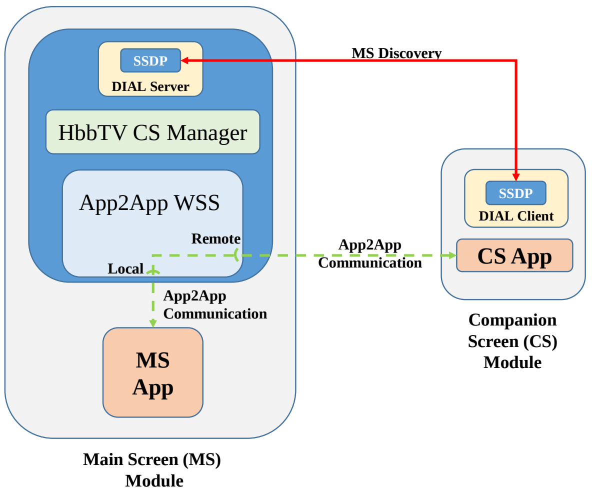


Fig. 9. Components and communication functionalities involved in the platform

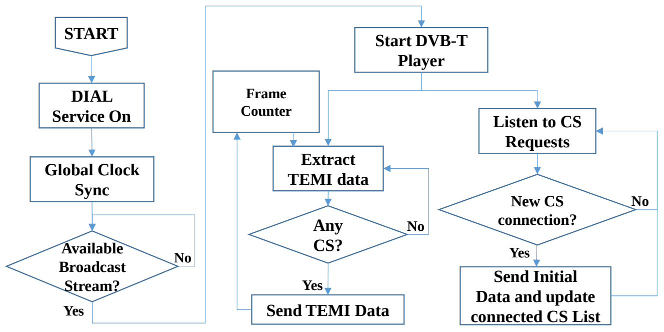


Fig. 10. Flow diagram of processes executed by the MS Module

C. Media Sync: Hybrid Sync and IDES

The platform provides intra-media sync for each particular media component as well as inter-media sync between the involved media components within each multiplexed incoming stream processed by each one of the GStreamer pipelines. These media sync types are provided thanks to the (relative) timelines included in the involved streams, and using native functionalities of GStreamer [16].

In addition, proper components and functionalities have been designed and developed to provide hybrid sync on a single device and/or on multiple devices, making use of a Master/Slave scheme ([10], [25]), in which the MS App acts as the Master and the CS App(s) act as the Slave(s). The initial steps to achieve hybrid sync, using the Master/Slave scheme, consist of: i) retrieving the TEMI timelines from the incoming MPEG2-TS, which give information about the generation instants of the video frames; ii) tracking the video frames until the rendering elements (i.e., the audio/video sinks); and iii) registering the "estimated" presentation (absolute, NTP-based) timestamps[30] of these video frames. The tuple of NTP-based generation timestamp (TEMI timeline) of the currently processed content and its "estimated" presentation timestamp will be sent (via App2App WSS) to the involved CS Apps (being executed on either the same hybrid terminal and/or on companion devices). The inclusion of the "estimated" presentation timestamps allows achieving higher sync accuracy than the inclusion of the reception timestamps as it allows overcoming variability issues, regarding both network and end-system delays and jitter. The frequency of the report intervals from the MS to the CS App(s) can be configured (e.g., one report every 2s or by using an incoming frame counter, see Figure 10).

Upon receiving these reports, each CS App will be able to compare its own playout timing with the one of the MS App. They will calculate the asynchrony (i.e., playout time difference) between them and perform the required playout adjustment (if needed). In order to control the period of the adjustments, and avoid too many continuous adjustments (which might be annoying to users or unnecessary), a second execution thread and a timer have been used (see Figure 11).

[30] The NTP-based 'estimated' presentation instant is calculated taking into account the delays of all the elements in the Gstreamer pipeline until the audio/video sinks or rendering elements (see Figure 8). The delay until the content is really presented to the output device (i.e., display or loudspeaker) is not taken into account in this version of the platform (left for future work).

To achieve a coherent notion of time in the session and to compensate for the effect of network delays between the MS and CS Apps, a common clock reference is used (provided by NTP, as explained in Section IV-A3).

The sync process is divided into two main processes (as in many other works, e.g. [25], [27] and [30]): initial (a.k.a. coarse) sync and continuous (a.k.a. fine) sync. The initial sync process is performed once launching the CS App and consists of seeking to a specific playout position in order to be in-sync with the playout process of the MS App. The continuous sync process is performed throughout the duration of the media session (e.g., every time a report from the MS App is received) and consists of regularly monitoring the asynchrony between the MS and CS Apps' playout processes (by comparing their playout timings). Then, if the asynchrony exceeds a specified threshold (e.g., 80ms), required adjustments to the playout process of the CS App will be performed to achieve sync. In this process, proper playout buffering policies have been adopted to more efficiently compensate for delay differences. Likewise, two types of playout adjustment techniques can be adopted [51] [52]: aggressive and smooth. On the one hand, aggressive techniques, such as skips and pauses, are employed when the asynchrony is higher than an upper threshold in order to achieve a nearly immediate sync process. On the other hand, Adaptive Media Playout (AMP) techniques can be employed when the asynchrony is within (configurable) limits. AMP consists of smoothly adjusting (i.e., fasting up or slowing down) the playout rate to correct asynchrony situations (minimizing them). It allows achieving higher sync accuracy and avoiding long-term playout disruptions, which can be annoying to the users' perception (bad Quality of Experience, QoE) [52].

The flow diagram of the different processes executed by the CS App to achieve an in-sync playout and their interactions is shown in Figure 11. As soon as the CS Module discovers any available MS Module, it receives initial data (such as the URIs to the available extra contents and the global clock reference in use) from that MS Module. With this information, the CS App can start playing out the related contents. During the playout process, the CS App will periodically receive the aforementioned MS App's playout timing information, which it will use to compare with the timing of its own playout process. If the (configurable) asynchrony threshold is exceeded, then playout adjustments will be performed. Although the asynchrony is calculated for the video components, as the TEMI timelines are inserted for each I-frame, the playout adjustments are performed to the video and audio branches of the pipelines of the involved CS Apps, thus also guaranteeing inter-media sync. As it can be seen in Fig. 11, two parallel threads were programmed in order to achieve IDES. The reason was because it resulted in a much better performance than when using a single thread, minimizing blocking an freezing effects during the playout.

This behavior loops until the end of the media session, and it is restarted every time a new media content is selected for being played out.

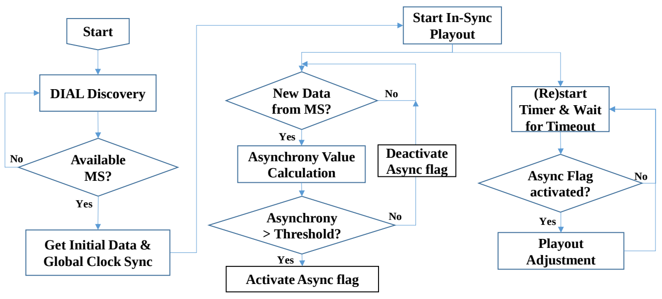


Fig. 11. Flow Diagram of the Processes Executed by each CS App

V. Evaluation

As mentioned, although the platform can provide support for many of the hybrid media services introduced in Section I, a typical one has been selected and implemented to objectively and subjectively evaluate it. The evaluation scenario, methodology and some obtained results are presented in this section.

A. Evaluated Scenario

The platform has been evaluated for a multi-view multi-device TV use case, which has the potential of providing highly immersive and personalized TV consumption experiences, as reflected in [6]. An overview of the evaluated scenario and use case is provided in Figure 12. Four different camera views from different angles of a ‘book reading’ event have been recorded at the TV studio of our university[31]. The four views are from four different cameras: front camera, front left and front right cameras, and right side camera. Although other genres, such as sports (e.g., football, Moto GP or Formula 1) and concerts, undoubtedly have higher potential, we have not used them because of a lack of appropriate contents and copyright issues.

The initial idea was that each one of the views were dynamically selected by the director/controller and delivered via broadcast (DVB-T). However, in the evaluated scenario, the 4 camera views have been multiplexed as 4 Elementary Streams (ES) in the same MPEG2-TS, which is transmitted via DVB-T in a single UHF (Ultra High Frequency) channel. The TS will be received and de-multiplexed, and one of the camera views will be selected and played out by the MS App on the hybrid terminal (see Figure 13). All the 4 views can also be delivered via broadband (DASH), and be played out by CS Apps running on the same hybrid terminal and/or on companion device/s (see Figure 12). The DVB-T content is encoded, encapsulated and delivered, as explained in Section IV.A.1, while the DASH contents are prepared and stored on a web server, as explained in Section IV.A.2.

When being played out on the hybrid terminal, the broadband content(s) can be presented in a mosaic view, in full-screen mode (as in Figure 13), or even in PiP mode (Figure 14). A menu has been added to the developed HbbTV app (running on the hybrid terminal - MS App - and/or on the companion devices - CS App -) to be able to dynamically select, or switch between, the camera view(s) from which to experience the event (see the red dashed box in Figure 13, and Figures 14 and 15). Figure 16 shows a complete scenario for the consumer’s side, in which the MS App running on the hybrid terminal (main TV) is playing out the (broadcast) DVB-T content, and four CS Apps running on different companion devices are playing out the (broadband) DASH contents, providing different views of the event.

To visually check the sync accuracy that is achieved, a text overlay with the frame number was added to each video frame during the encoding process (by using *ffmpeg*). Likewise, to visually check the quality of the DASH (and HLS) contents being played out by the CS App, a text overlay was added to each one of the segments of all the different qualities, indicating its bitrate and resolution (see Figures 14 and 15).

Tables III and IV show the main features of the hybrid contents and the different components and devices used in the evaluation scenario, respectively.

Only DASH contents have been used for the broadband delivery of the available camera views (as it is the adopted technology by HbbTV 2.0.1). The XML file used for the evaluated scenario is presented in Figure 17. For each view, a timestamp has been included in the file (*temi_init* parameter) in order to more effectively achieve the sync goal. It represents the global generation timestamp corresponding to the first video Media Unit (i.e., video frame) of the delivered content. The file also includes the address of the NTP server of our Spanish national research and education network (Iris Network), *ntp.redimadrid.es*. It has been used by all the involved devices to insert (and interpret) the NTP-based timelines.

TABLE III. Hybrid Contents

Technology	Content	Encoding
Broadcast (MPEG2-TS)	Video	H.264, 1920x1080, 25fps (4 channels, 30Mbps)
	Audio	MPEG-4 AAC 140Kbps (2 channels, 48KHz)
Broadband (DASH)	Video profile	H.264, 25 fps, duration of segments: 3s Quality 1 (Q1): 426x240, 700Kbps Q2: 640x360, 1000Kbps Q3: 854x480, 2000Kbps Q4: 1280x720, 4000Kbps Q5: 1920x1080, 6000Kbps
	Audio	MPEG-4 AAC 128Kbps (2 channels, 48KHz)

TABLE IV. Components and Devices Used in the Evaluated Scenario

PC (*broadcaster*)	Intel Core i7-6700 @ 3.40GHz, 8GB RAM, SSD 240GB, Windows 10, DVB-T PCI Card: DekTec DTA-2111
PC (server)	Intel Xeon E5420 @ 2.50GHz x8, 8GB RAM, HDD 200GB, Ubuntu 14.04, Apache Server v2.4.7, Fast Ethernet embedded card.
Switch/Router	TP-Link AC1900 Wireless Dual Band Gigabit Router. IEEE 802.11ac.
MS (Set Top Box, STB)	PC1: Intel Celeron 1037U @ 1.80GHz x2, 4GB RAM, HDD 150GB, Ubuntu 14.04; Fast Ethernet embedded card; WiFi card IEEE 802.11 b/g/n; DVB-T USB card: Hauppage! Nova-T Stick 3; TV LG 32LF592U (32”).
CS	PC1, Samsung Galaxy Tab S (IEEE 802.11ac 10” *tablet*) and Samsung S5 (5” *smartphone*)

[31] The evaluation scenario is based on recorded and stored contents. The evaluation for live contents is left for future work.

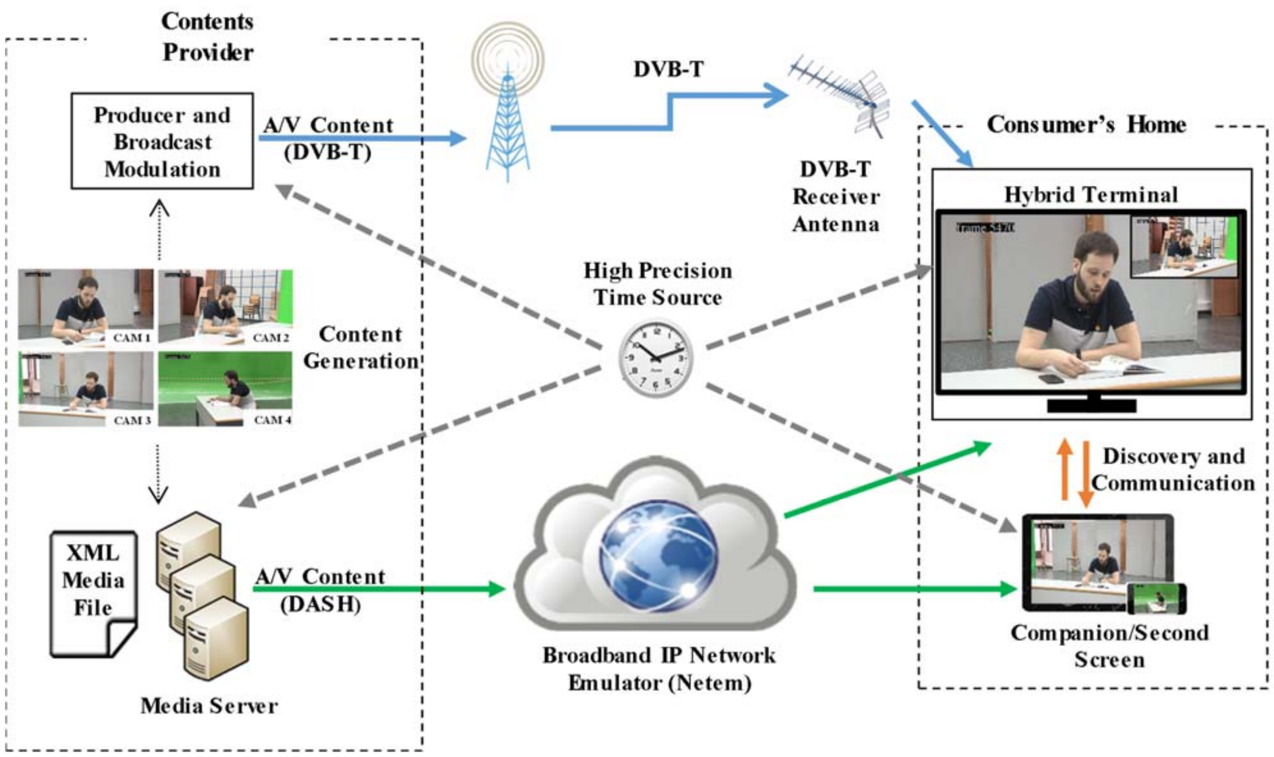


Fig. 12. Overview of the use case and scenario used for the evaluation of the platform

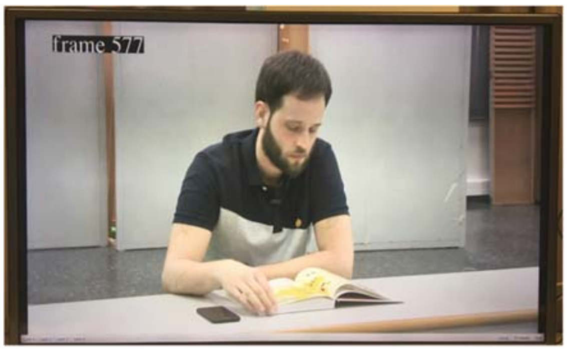


Fig. 13. MS App running on the hybrid terminal (main TV), with 4 buttons to select each channel (camera view)

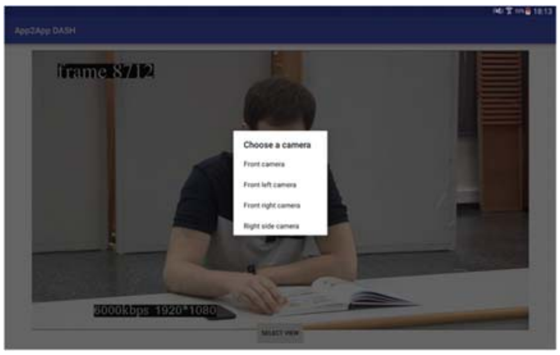


Fig. 15. Menu for dynamically selecting the available related hybrid media contents on companion devices

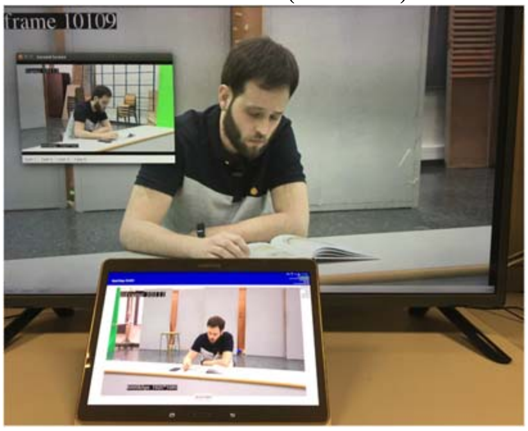


Fig. 14. Playout of hybrid media contents in a Multi-Screen scenario with 3 different views (Linux-based Hybrid Terminal with MS and CS Apps, in PiP mode, and Android-based tablet with CS App)

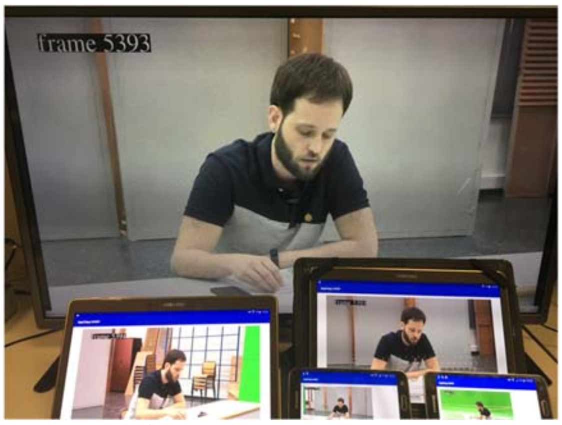


Fig. 16. Playout of hybrid media contents in a Multi-Screen scenario

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Hybrid Media Contents File>
<MEDIA id="1" media_type="AV" media_format="h264/aac"
            metadata="front_camera/spanish"
            temi_init="3699255471291907022">
   <source protocol="http/dash"
         uri="http://IP_Server/multicam_scenario/cam1/stream.mpd"/>
   <source protocol="rtsp" uri="rtsp://IP_Server:8551/test"/>
</MEDIA>
<MEDIA id="2" media_type="AV" media_format="h264/aac"
            metadata="front_left_camera/spanish"
            temi_init="3699255471291907022">
    <source protocol="http/dash"
          uri="http://IP_Server/multicam_scenario/cam2/stream.mpd"/>
    <source protocol="rtsp" uri="rtsp:// IP_Server:8552/test"/>
</MEDIA>
<MEDIA id="3" media_type="AV" media_format="h264/aac"
            metadata="front_right_camera/spanish"
            temi_init="3699255471291907022">
   <source protocol="http/dash"
         uri="http://IP_Server/multicam_scenario/cam3/stream.mpd"/>
   <source protocol="rtsp" uri="rtsp:// IP_Server:8553/test"/>
</MEDIA>
<MEDIA id="4" media_type="AV" media_format="h264/aac"
            metadata="right_side_camera/spanish"
            temi_init="3699255471291907022">
   <source protocol="http/dash"
         uri="http://IP_Server/multicam_scenario/cam4/stream.mpd"/>
   <source protocol="rtsp" uri="rtsp:// IP_Server:8554/test"/>
</MEDIA>
<CLOCK id="1" protocol="ntp" media_type="time"
            media_format="64_bit_ntp_time"
            metadata="" uri="ntp.redimadrid.es"/>
<LASTUPDATE protocol="http" media_type="time"
            metadata="" format="dd/mm/yyyy-hh:mm:ss"
            value="11/04/2017-11:20:00"/>
</Hybrid Media Contents File>
```

Fig. 17. XML file for the evaluated scenario

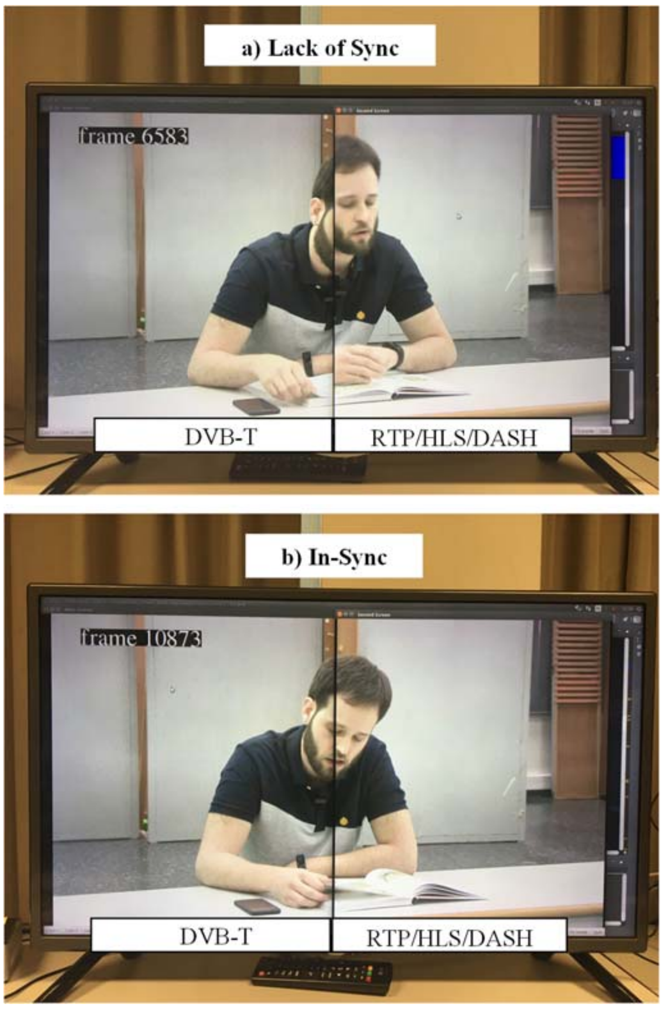


Fig. 18. Video cropping to visually check the sync accuracy

A list of demo videos showing the capabilities and performance of the platform for this use case (when using DASH, and also HLS or RTP as broadband technologies), but for a multi-view music concert, is available at: http://bit.ly/2rmjTYM.

B. Evaluation Methodology

Although lip-sync and audio echo issues can be perceived when playing out the hybrid media contents in a PiP and in a mosaic view, it is difficult to visually assess the level of sync accuracy that is achieved in these configurations.

A practical method to check it consists of receiving the same media content both via broadcast (DVB-T) and via broadband (DASH, HLS or RTP), cropping the two video streams into the left and right half parts, and placing them one next to the other. If both videos are perceived as a single video (Figure 18b) and audio sounds of both match (i.e., they cannot be distinguished, or no echoes are heard), it means that the obtained sync levels are very satisfactory. Otherwise, un-alignments between the videos (see Figure 18a) or audio echoes will be a sign of a lack of sync.

Likewise, another more precise method to visually check the sync accuracy, even in multi-screen scenarios, consists of adding frame number overlays to the video frames during the encoding processes and comparing them during playout. It can be easily done by recording a video (of the multi-view and multi-device scenario) and pausing it and/or by taking pictures at specific moments. Figure 14 shows a picture of the in-sync multi-screen scenario with a difference of 2 frames[32] (frame numbers 10109 in the MS App and 10111 in both CS Apps) between the playout of the MS App and 2 CS Apps (one running on the hybrid terminal and another on a secondary device). The previously provided link includes demo videos showing this kind of evaluation tests.

Two more automatic and objective asynchrony measurement methods can additionally be employed. The first one consists of registering the asynchrony between the involved contents at the stream level, by comparing the (relative and absolute) timelines of their video frames being played out. In addition, if all the involved streams contain the

[32] In that scenario the video frame rate was 25 frames/s, so a difference of 2 frame means an asynchrony of 80ms (2*1/25). Also, note that an asynchrony threshold of 80ms was set, so it means that no playout adjustments are performed until the detected asynchrony exceeds that threshold.

same audio, a second method to more accurately measure the sync accuracy would consist of capturing their audio outputs (e.g., the right channel from the MS and the left channel from the CS), recording them and computing the cross-correlation between their samples. In this work, we have followed the former method. The implementation and use of the latter method is left for further work.

In the next section, the obtained results are presented.

C. Objective Evaluation

The performance of the platform's functionalities, paying special attention to the sync accuracy, has been objectively evaluated in different configurations (wired and wireless scenarios) and under different conditions (in ideal and more realistic scenarios, i.e., forcing specific network conditions).

The frequency of the report intervals sent by the MS Module to the CS Module(s) was configured to one report every 15[33] incoming frames (i.e., every 0.6s for the used video sequences, with frame rates of 25 fps). The value of the asynchrony threshold between the playout processes of the MS and CS Apps was set to ±80ms (i.e., a difference of 2 frames in a video with a frame rate of 25 fps). The magnitude of this value, although slightly exceeds the frame-accuracy boundaries, falls within the tolerable ranges in this kind of scenarios ([53]), provides satisfactory performance results, and results in unnoticeable asynchrony levels (as proved in Section V.D).

All the experiments were conducted 10 times, and the Root Mean Square (RMS) value[34] of the measured asynchrony and the 95% Confidence Interval (C.I.) level are presented in tables. In this paper, the range and distribution of asynchrony and playout adjustments values are not considered (left for future work).

The following subsections present the results obtained in an ideal scenario and, then, in more realistic scenarios, considering/forcing loses, delays and BW limitations.

1) Ideal Scenario

In this case, there were no forced packet losses in DVB-T, and no forced delays and BW limitations in the broadband (IP) network. Two different scenarios were evaluated: a) the MS and CS Modules running on the same hybrid terminal (e.g., Smart TV), and b) the MS Module running on the hybrid terminal and the CS Module running on a tablet and a smartphone (with WiFi connection). The employed components and devices are listed in Table IV.

1a) MS and CS Modules running on the (same) hybrid terminal

In this case, two experiments were conducted, depending on the connection type of the hybrid terminal to the home network WiFi router: wired or wireless. In the former, the hybrid terminal was directly connected with a twisted pair cable to the 100 Fast Ethernet (100Mbps) switch included in the router. In the latter, the hybrid terminal was connected to the router through a 72 Mbps IEEE 802.11 WiFi connection. The HTTP server with DASH contents was directly connected via a twisted pair cable to the 100 Fast Ethernet (100Mbps) switch of the home network WiFi router.

Table V summarizes the obtained results (RMS value of the asynchrony together with 95% C.I. levels). As it can be observed, the RMS value of the asynchrony was significantly lower than the 80ms threshold for both experiments.

TABLE V. RMS VALUE OF ASYNCHRONY FOR THE IDEAL CASE ON A SINGLE DEVICE (HYBRID TERMINAL)

MS and CS Modules running on the Hybrid Terminal	
Wired Terminal	Wireless Terminal
31,26±7,94ms	38,51±10,01ms

1b) MS and CS Modules running on different devices

In this case, the MS and CS Modules were executed on different devices: the MS Module on a wired Linux-based device (hybrid terminal) and the CS Module on a wireless 10" and 5" Android-based tablet and smartphone, respectively. The obtained results are presented in Table VI. In this case, the RMS value of asynchrony was also significantly lower than the configured threshold (80ms). As expected, higher sync accuracy and better performance was achieved when using the tablet, as it has better processing resources.

TABLE VI. RMS VALUE OF ASYNCHRONY FOR THE IDEAL IN A MULTI-SCREEN SCENARIO

MS and CS Modules running on different devices	
Tablet	Smartphone
43,61±11,60ms	56,62±11,81ms

2) More realistic scenario with loses and BW limitations

In this case, three tests were conducted:

- T2.1: only with packet loss in broadcast connection, but no BW limitations and no delays were forced in the broadband connection;
- T2.2: only with BW limitations and delays in broadband connection, but without forcing packet loss in the broadcast connection;
- T2.3: with forced packet loss in the broadcast connection as well as BW limitations and delays in the broadband connection.

The DekTec StreamXpress software configuration tool, provided with the DekTec DTA-2111 PCI card, has been used to force packet losses in the broadcast connection, while the Netem network emulator[35] has been used to force BW limitations and network delays in the broadband connection.

On the one hand, regarding DVB broadcasting, each of the MPEG2-TS transport packets are usually extended by a shortened Reed-Solomon error protection code, leading to a DVB MPEG2-TS packet with a length of 204 bytes. In combination with convolution coding and appropriate

[33] For lower values, the performance of the secondary devices does not improve, according to preliminarily conducted tests.

[34] We use root square values instead of mean values because the asynchrony values can be positive or negative, so mean values are not meaningful.

[35] https://wiki.linuxfoundation.org/networking/netem (last access, October 2017)

modulation schemes, a so called quasi-error free (QEF) transport of DVB services can be guaranteed, which means that, in average, only one non-correctable error occurs within one hour of program presentation (equivalent to a BER of $1x10^{-11}$). Taking into account that the videos used in the evaluation are around 2-minute long, we have evaluated the scenario with a symbolic packet loss probability of 0,05% (much larger than the QEF average) in DVB transmission.

On the other hand, regarding broadband delivery, in order to force quality switching (i.e., the selection of all the different versions or qualities of the content) in the DASH client integrated with the CS App, a particular pattern for changing the BW of the broadband connection was used in each one of the tests. Concretely, the BW of the network interface of the HTTP server was limited to 900 kbps, 1700kbps, 2300kbps, 4100kbps, and without limitation, in this order, changing it every 25 seconds (according to the bitrate of the generated DASH qualities, see Table III). Moreover, the network delay parameter was set to 60ms ± 20ms (following a normal distribution), which corresponds to what can be observed within long-distance fixed line connections or reasonable mobile networks and, thus, is representative for a broad range of application scenarios. Both parameters (BW and delay) were controlled by using the Netem tool.

Next, the results for the three tests (T2.1, T2.2 and T2.3) are presented for both a single device scenario (MS and CS Apps running on the same hybrid terminal) and a multi-device scenario (MS Module running on the hybrid terminal and CS Module running on a tablet and smartphone).

2a) MS and CS Modules running on the hybrid terminal

The obtained results when both the MS and CS Modules were executed on the (same) hybrid terminal for the three tests are presented in Table VII, when the CS App accesses the broadband contents via both the wired and wireless connection. It can be observed that the RMS value of the asynchrony was also lower than the configured threshold.

TABLE VII. ASYNCHRONY VALUES FOR A MORE REALISTIC CASE WHEN USING A SINGLE DEVICE (HYBRID TERMINAL)

MS and CS Modules running on the same Hybrid Terminal		
Test	Wired Connection	Wireless Connection
T2.1: Packet loss of 0.05% in broadcast connection	54,54±11,63ms	58,94±13,48ms
T2.2: BW limitations and delays in broadband connection.	48,07±10,30ms	49,34±11,87ms
T2.3: Packet loss of 0.05% in broadcast connection together with BW limitations and delays in broadband connection.	56,59±12,87ms	60,56±13,61ms

2b) MS and CS Apps running on different devices

The obtained results when the MS and CS Modules were executed on different devices for the three tests, when using both a tablet and a smartphone as companion devices, are presented in Table VIII. In this case, it can also be observed that, although the RMS value of the asynchrony was a bit higher than in the previous case, it is still lower than the configured threshold. Similarly to the ideal case, the sync accuracy for the smartphone was not as good as for the tablet, due to its poorer performance and lower processing capabilities/resources. Indeed, the RMS value of the asynchrony might exceed the threshold for the T2.3 test when using the smartphone. However, the obtained results are still acceptable in such a case, especially taking into account the quite bad forced network conditions and the limited performance of the smartphone.

TABLE VIII. ASYNCHRONY VALUES FOR A MORE REALISTIC CASE IN A MULTI-DEVICE SCENARIO

MS and CS Modules running on different devices (MS on a Wired Terminal CS App on a WiFi device)		
Test	Tablet	Smartphone
T2.1: Packet loss of 0.05% in broadcast connection	61,05±13,17 ms	64,61±13,14 ms
T2.2: BW limitations and delays in broadband connection.	51,93±12,58 ms	58,91±12,20 ms
T2.3: Packet loss of 0.05% in broadcast connection together with BW limitations and delays in broadband connection.	61,69±14,10 ms	77,88±15,78 ms

3) Discussion

In all the conducted tests, the RMS value of the asynchrony was kept significantly below the configured threshold of ±80ms. After an initial sync process when launching the CS App, the asynchrony was kept within that configured threshold most of the time. If a different camera view is selected, sync is soon recovered by performing the proper playout adjustments. Despite the existence of some low-range fluctuations in the asynchrony value, the asynchrony was kept within the allowable thresholds most of the time, except for some sporadic situations (e.g., due to packet loss, the launching of the CS App, when switching to a different camera view…). Moreover, those fluctuations had not an impact on the fluidity and smoothness of the playout process, as demonstrated with the results of the subjective assessment (in next sub-section).

As it can be observed in Tables V and VII, the obtained RMS values of the asynchrony were a bit higher when the hybrid terminal was wirelessly connected to the home router than when it was connected via a twisted pair cable, as expected.

RMS values of the asynchrony were also a bit higher when the CS App was run on a companion device than when it was ran on the hybrid terminal (i.e. in the same device than MS App), and a bit higher when using the smartphone than when using the table, as expected.

The worst (i.e., highest) values were obtained when both packet loss in the broadcast connection and BW limitations and delays in broadband connection were forced (T2.3). The obtained values in T2.3 were worse than in T2.2, but very similar to T2.1, which means that, during the media session, packet loss in broadcast connection affected in a more significant manner to the performance of hybrid sync than the configured broadband limitations. In the latter case, such limitations were well managed by the HTTP/TCP connection and by adequate adaptive buffering and quality switching techniques implemented in the DASH player.

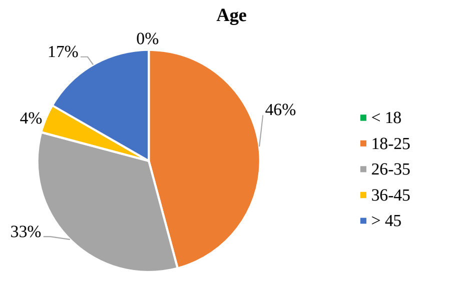
As shown in this section, the hybrid sync solution implemented in the presented platform is able to keep the mean values of asynchrony below the configured threshold, in both single- and multi-device, wired and wireless, scenarios, including when forcing quite bad network conditions and using devices with limited processing resources.

D. Subjective Evaluation

The success of a multimedia platform or application is mainly determined by its acceptability by the end users (consumers). This acceptability, in turn, will depend on many aspects, such as the usability (i.e., the ease of use) of the platform, its GUI design, performance, the usefulness and benefits of the functionalities it provides, and its applicability in scenarios of interest. In order to test these aspects, a subjective evaluation study was conducted, in which 24 users participated. 62% of those participants were men, and also 62% had a technological background. Their age distribution and level of studies are provided in Figure 19.

The tests began with a brief introduction and contextualization of the research topic and goal. Next, the participants were asked to experience with the platform, by testing its functionalities for switching between the 4 available multi-view videos, when using two types of companion devices (tablets and smartphones). After each test, they had to fill in a specific section of a questionnaire. Next, the results of the study are summarized.

The first test was focused on assessing the relevance of IDES between the hybrid terminal and a companion device. For that purpose, the participants experienced with the platform, with a 48" TV screen (running the MS Module), and a 10" tablet (running a CS Module) playing out the same video[36] (front camera). 5 different cases in which different asynchrony values between the playout processes of the involved devices were forced (presented in Table IX).



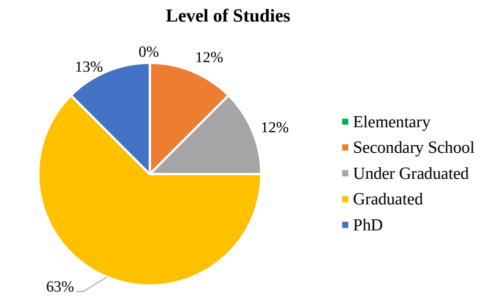


Fig. 19. Participants' Data

[36] When both devices display the same view, asynchrony situations are more easily noticed by users than when they display different views. So, we have selected the most restrictive case for the assessment.

TABLE IX. DIFFERENT ASYNCHRONY CASES

Case	Description
1	Asynchrony: -3s (CS App's playout lagged 3s).
2	Asynchrony:-1s (CS App's playout lagged 1s)
3	Asynchrony: 0s (no forced asynchrony).
4	Asynchrony: +1s (CS App's playout advanced 1s).
5	Asynchrony: +3s (CS App's playout advanced 3s).

Notice that Case 3 is used to test the adequate performance of the platform (i.e., if the users perceived a highly accurate sync between the playout of the hybrid contents in the multi-device scenario, when no asynchrony was forced).

The 5 cases were presented to the users in a random order. They watched the videos in each case during 3 minutes to assess the level of perceptibility and/or annoyance of each asynchrony value, using the Mean Opinion Score (MOS) metric (see Table X).

TABLE X. MEAN OPINION SCORE (MOS) SCALES

MOS	**Quality**	**Impairment**
5	Excellent	Imperceptible
4	Good	Perceptible, but not annoying
3	Fair	Slightly annoying
2	Poor	Annoying
1	Bad	Very annoying

The obtained results are shown in Figure 20, including mean values and 95% C.I. levels. It can be seen that the perceived QoE was close to 5 (i.e., Excellent) when there was no forced asynchrony (i.e., Case 3), and that it dropped when the asynchrony become higher (especially when it was 3s, in Cases 1 and 5). These results confirm the relevance of providing highly accurate IDES solutions in multi-screen scenarios, such as the one designed in this work.

Next, the users were asked to freely experience with the platform without forced asynchrony values, when using both tablets and smartphones as companion devices, while watching the broadcast content on the main TV, and switching between the available camera views on those devices. After that, they had to fill in the remaining parts of the questionnaire.

Similarly, as before, they were asked about their perception and satisfaction about the obtained sync levels. In this case, also a 5-level Likert-type scale was used to indicate the level of agreement regarding the statement of the title of Figure 21, with the 5 levels shown in that figure as the possible answers. As it can be seen, 58% of them completely agreed with the assertion that sync between the playout processes of the main TV and the companion device was accurate, while 38% partially agreed (the remaining 4% was neutral). Indeed, the participants were also asked to rate the perceived sync accuracy by using a 1-10 scale, and the mean score was 9.00 (i.e., close to perfect).

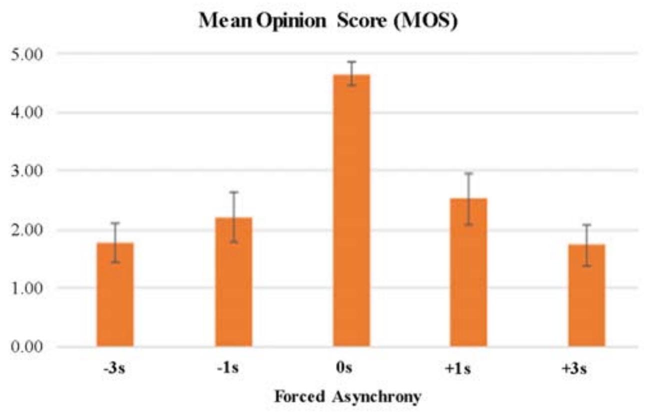


Fig. 20. Impact on the QoE of different asynchrony levels for IDES

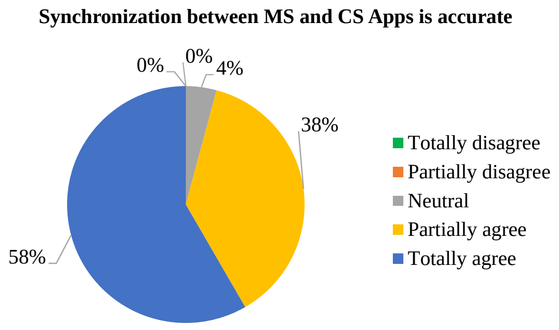


Fig. 21. Perceived Sync Accuracy between MS and CS

The participants were also asked about the acceptability or annoyance (i.e., their perception about detected impairments, see table X) of the delays when both launching the CS and when switching to a different camera view. As it can be seen in Figure 22, although the delays were noticed by a significant percentage of participants, they did not consider their magnitudes as annoying, which can be considered as quite satisfactory results.

Additionally, the participants were also asked about their consumption habits, their previous experience in this kind of scenarios and applications, the awaken interest, and the usability and usefulness of this platform. As it can be seen in Figure 23, a significant percentage of them declared to use companion devices to consume additional contents (e.g., to access to extra information) while watching TV. The genres with higher demands regarding extra contents were sports (19% of participants), music videos (17%), series (11%) and shows (11%). 25% of participants affirmed having had a previous experience with similar platforms using companion devices for consuming extra content. 67% of them declared having experienced problems with these platforms, mainly due to high delays and lack of sync, which resulted annoying to them.

As reflected in the top part of Figure 24, most of the participants thought that it is easy and comfortable to use the implemented platform (including both the MS and CS Apps). Only 4% of them partially disagreed with that assertion. Likewise, they mostly agreed that it is an interesting and useful platform (bottom part of Figure 24). Most of them believed that this platform can provide more immersive, personalized and enriched TV watching experiences (83% totally agreed; 11% partially agreed; and the rest were neutral with that assertion). Moreover, 96% of them considered that this platform can have a big impact in the media consumption market (79% totally agreed; 17% partially agreed; and 4% were neutral with that assertion).

All the participants declared interest in using this platform, if it were available.

Therefore, the presented results reflect the satisfactory performance and usability of the platform, the awaken interest, and its potential impact in the TV consumption landscape.

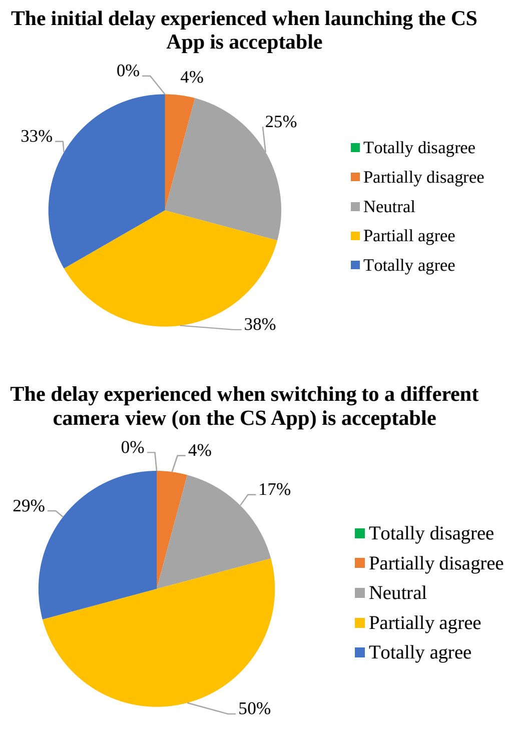


Fig. 22. Perception of Delays when using the Platform

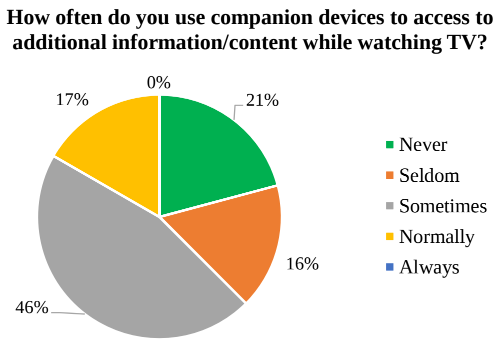
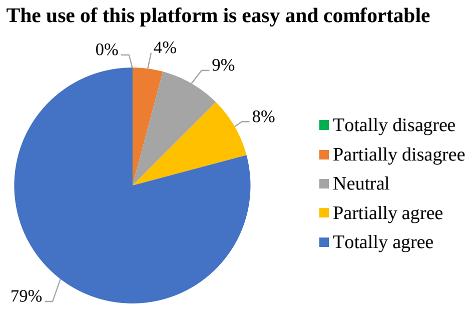


Fig. 23. Use of Companion Devices while watching TV



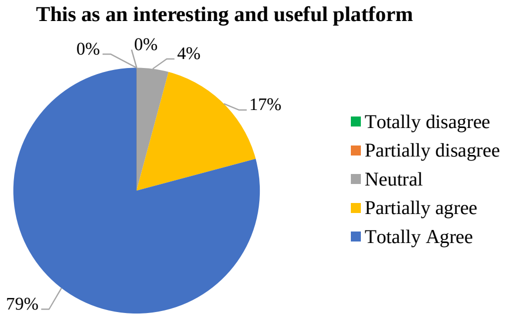


Fig. 24. Usability and Usefulness of the Platform

VI. Conclusions & Future Work

After summarizing some key concepts, components and technologies for hybrid media delivery and synchronized consumption, this paper has presented a prototype of an end-to-end HbbTV-compliant platform for the preparation, (adaptive) delivery and synchronized playout of related hybrid (broadcast and broadband) contents. These contents can be played out on a single device (e.g., connected or Smart TVs) and/or on different close-by devices (e.g., multi-screen scenarios). The platform supports the key features of the latest HbbTV release (v2.0.1) and additional ones that are necessary to successfully deploy hybrid and multi-device TV services.

These additional ones are: 1) signaling mechanisms to discover, describe and associate the available related (hybrid) media contents; 2) interaction and coordination mechanisms between the available consumptions devices; and 3) adaptive sync solutions (including protocols, algorithms and adjustment techniques) to accurately time-align the consumption of the related media contents. The platform can provide support for many hybrid media services, such as the ones reviewed in Section I. Nevertheless, it has been objective and subjectively evaluated for the use case of multi-view and multi-device TV, which is a very relevant use case [6], obtaining promising results in terms of stability, responsiveness, delays and sync accuracy.

Likewise, the results of the subjective evaluation reveal that users were very satisfied with the performance and usability of the platform for the implemented use case, as well as with the usefulness of its functionalities. Moreover, the users declared being very interested in this kind of hybrid media services. It reflects the commercial potential of the platform and its relevance to enable more interactive, immersive, personalized and lean-forward TV watching experiences, opening the door to a new wave of enriched and fascinating services.

The development of the platform is still not finished. For instance, the launch of the CS App(s) is manually done by users, but the dynamic CS App launch by the MS App, via the DIAL protocol, will soon be integrated. Moreover, in order to have a fully GStreamer-based platform, a custom GStreamer element to insert the TEMI descriptors and AIT data into MPEG2-TS streams will be also developed. In addition, at this stage, each GStreamer pipeline of the platform can process one audio and one video component. The handling of additional audio and video components and of other types of media components, such as subtitles and teletext, will be included in a future release.

Further work will also be targeted on minimizing delays (including the different steps along the end-to-end delivery chain, when launching the CS App and when switching to a different camera view), on achieving higher sync accuracy (e.g., frame-accurate sync) and on adopting proper playout adjustment techniques to maximize the perceived QoE [52]. We also plan to adopt event-driven reporting rather than regular reporting features, as in [54], which will result in a better performance and will enable a more efficient implementation of advanced interactive use cases (e.g., quizzes, gaming …). Currently, the platform is being extended to provide support for IDMS. This will enable interactive and collaborative shared media experiences between geographically distributed consumers. In addition, the platform will be extended to provide support for other types and formats of media contents (e.g., Ultra High Definition or UHD, omnidirectional contents…) as well as live contents. Finally, the platform will be implemented and tested in real TV systems, rather than in lab-controlled scenarios.

Acknowledgment

Authors would like to thank “Samsung Electronics Iberia, S.A” for providing the consumption devices (connected TVs, tablets and smartphones) for testing the prototypes and conducting the users’ studies.

References

[1] A. Begen, T. Akgul, and M. Baugher, “Watching Video over the Web: Part 1: Streaming Protocols,” *IEEE Internet Computing*, vol. 15, no. 2, pp. 54-63, Apr. 2011.

[2] RTP: A Transport Protocol for Real-Time Applications, IETF Internet Standard, RFC 3550, 2003, [Online]. Available: https://tools.ietf.org/html/rfc3550

[3] HTTP Live Streaming, IETF Internet Draft, 2003, [Online]. Available: https://tools.ietf.org/html/draft-pantos-http-live-streaming-20

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

[5] Hybrid Broadcast Broadband TV (HbbTV) 2.0.1 Specification, HbbTV Association Resource Library, 2016, [Online]. Available: https://www.hbbtv.org/resource-library

[6] F. Boronat, M. Montagud, D. Marfil, and C. Luzón, “Hybrid Broadcast/Broadband TV Services and Media Synchronization. Demands, Preferences and Expectations of Spanish Consumers,” *IEEE Transactions on Broadcasting*, in press (2017).

[7] J. Le Feuvre. (2015, June). A Test Bed for Hybrid Broadcast Broadband Services. Presented at MediaSync Workshop 2015, Brussels, Belgium, [Online]. Available: http://bit.ly/2hbdibP

[8] R. van Brandenburg, and A. Veenhuizen. (2013, October). Immersive second-screen experiences using hybrid media synchronization. Presented at MediaSync Workshop 2013, Nantes, France, [Online]. Available: http://bit.ly/2zj5BLL

[9] M. O. Van Deventer, H. Stokking, O. A. Niamut, F. A. Walraven, and V. B. Klos, "Advanced Interactive Television Service Require Synchronization," in Proc. 15th International Conference on Systems, Signals and Image Processing (IWSSIP), Bratislava, Slovak Republic, 2008, pp. 109-112.

[10] M. Montagud, F. Boronat, H. Stokking, and R. van Brandenburg, "Inter-Destination Multimedia Synchronization; Schemes, Use Cases and Standardization," *Multimedia Systems Journal*, vol. *18*, no. 6, pp. 459-482, Nov. 2012.

[11] W. Kooij, H. M. Stokking, R. van Brandenburg, and P-T. de Boer. (2014, June). Playout delay of TV signals: measurement system design, validation and results. Presented at ACM TVX 2014, Newcastle (UK), [Online]. Available: https://research.utwente.nl/files/5442732/p23-kooij.pdf

[12] L. Beloqui, F. Boronat, M. Montagud, and H. Melvin, "Understanding Timelines within MPEG Standards," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 1, pp. 368-400, 2016.

[13] Network Time Protocol Version 4: Protocol and Algorithms Specification, IETF Internet Standard, RFC 5905, 2010, [Online]. Available: https://tools.ietf.org/html/rfc5905

[14] IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, IEEE 1588-2008, 2008.

[15] N. Borch, F. Daoust and I. Arntzen (2016). Timing-small step for developers, giant leap for the media industry. Presented at International Broadcasting Convention, IBC, 2016, Amsterdam RAI, Europaplein, NL 1078 GZ Amsterdam, The Netherlands, [Online]. Available: https://www.ibc.org/download?ac=720

[16] GStreamer framework, [Online]. Available: https://GStreamer.freedesktop.org/

[17] Specification for the carriage of synchronized auxiliary data in DVB transport streams, ETSI TS 102 823 V1.2.1, 2014.

[18] M.O. van Deventer, H. Stokking, M. Hammond, J. Le Feuvre and P. Cesar, "Standards for Multi-stream and Multi-device Media Synchronization," *IEEE Communications Magazine*, vol. 54, no. 3, pp.16-21, 2016.

[19] Delivery of Timeline for External Data, ISO/IEC 13818-1:2013/PDAM 6, 2013, [Online]. Available: http://bit.ly/2Aeog8i

[20] SDP: Session Description Protocol, IETF Internet Standard, RFC 4566, 2006, [Online]. Available: https://tools.ietf.org/html/rfc4566

[21] DIAL (DIscovery And Launch) protocol specification, Version 1.7.2, 2015, [Online]. Available: http://www.dial-multiscreen.org/dial-protocol-specification/DIAL-2ndScreenProtocol-1.7.2.pdf

[22] Companion Screens and Streams; Part1: Concepts, roles and overall architecture, ETSI TS 106 286-1, 2015, [Online]. Available: https://www.dvb.org/resources/public/standards/a167-1_dvb-css_spec.pdf

[23] Companion Screens and Streams; Part 2: Content Identification and Media Synchronization, ETSI TS 106 286-1, 2015, [Online]. Available: http://www.etsi.org/deliver/etsi_ts/103200_103299/10328602/01.01.01_60/ts_10328602v010101p.pdf

[24] V. Vinayagamoorthy, R. Ramdhany, and M. Hammond, "Enabling Frame-Accurate Synchronised Companion Screen Experiences," in Proc. ACM TVX'16, New York, NY, USA, 2016, pp. 83-92.

[25] F. Boronat, J. Lloret, and M. García, "Multimedia group and inter-stream synchronization techniques: A comparative study," *Information Systems*, vol. 34, no. 1, pp. 108-131, Mar. 2009.

[26] Z. Huang, K. Nahrstedt, and R. Steinmetz, "Evolution of temporal multimedia synchronization principles: A historical viewpoint," *ACM Trans. Multimedia Comput. Commun. Appl*., vol. 9, no. 1, article 34, 23 pages, Oct. 2013.

[27] L. Ehley, B. Furht and M. Ilyas, "Evaluation of multimedia synchronization techniques," in Proc. IEEE International Conference on Multimedia Computing and Systems (CMCS), Boston, MA, 1994, pp. 514-519.

[28] K. Matsumura, M. J. Evans, Y. Shishikui and A. McParland, "Personalization of broadcast programs using synchronized internet content," in Proc. International Conference on Consumer Electronics (ICCE), Las Vegas, NV., USA, 2010, pp. 145-146.

[29] M. Armstrong, J. Barrett, and M. Evans, "Enabling and enriching broadcast services by combining IP and broadcast delivery," BBC, London, UK, Research WHP 185, 2010, [Online]. Available: http://downloads.bbc.co.uk/rd/pubs/whp/whp-pdf-files/WHP185.pdf

[30] S. Aoki, K. Aoki, H. Hamada, Y. Kanatsugu, M. Yamamoto and K. Aizawa, "A new transport scheme for hybrid delivery of content over broadcast and broadband," in Proc. IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), Nuremberg, Germany, 2011, pp. 1-6.

[31] S. Aoki, K. Otsuki, and H. Hamada, "Effective usage of MMT in broadcasting systems," in Proc. IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), London, UK, 2013, pp. 1-6.

[32] S. Aoki, Y. Kawamura, K. Otsuki, N. Nakamura, and T. Kimura, "Development of MMT-based broadcasting system for hybrid delivery," in Proc. IEEE International Conference on Multimedia & Expo Workshops (ICMEW), Turin, Italy, 2015, pp. 1-4.

[33] Y. Lim, K. Park, J. Y. Lee, S. Aoki, and G. Fernando, "MMT: An Emerging MPEG Standard for Multimedia Delivery over the Internet," *IEEE Multimedia*, vol. 20, no. 1, pp. 80-85, 2013.

[34] L. Beloqui, S. M. Saleh Al-Majeed, H. Melvin, and M. Fleury, "Effective synchronization of Hybrid Broadcast and Broadband TV," in Proc. IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV., USA, 2012, pp. 160-161.

[35] L. Beloqui, and H. Melvin. (2012, October). Interactive Multi-source Media Synchronization for HbbTV. Presented at MediaSync Workshop, Berlin, Germany, [Online]. Available: http://bit.ly/2hKLX16

[36] C. Concolato, S. Thomas, R. Bouqueau, and J. Le Feuvre, "Synchronized Delivery of Multimedia Content over Uncoordinated Broadcast Broadband Networks," in Proc. ACM Multimedia Systems Conference (MMSys '12), Chapel Hill, North Carolina, USA, 2012, pp. 227-232.

[37] C. Köhnen, C. Köbel, and N. Hellhund, "A DVB/IP Streaming Testbed for Hybrid Digital Media Content Synchronization," in Proc. IEEE Second International Conference on Consumer Electronics - Berlin (ICCE-Berlin), Berlin, Germany, 2012, pp. 136-140.

[38] A. Veenhuizen. (2012, October). Frame accurate media synchronization of heterogeneous media sources in an HBB context. Presented at MediaSync Workshop, Berlin, Germany, [Online]. Available: http://bit.ly/2yB9dZX

[39] J. Le Feuvre, and C. Concolato. (2013, October). Hybrid broadcast services using MPEG DASH. Presented at MediaSync Workshop, Nantes, France, [Online]. Available: http://bit.ly/2y9Liwg

[40] GPAC Framework, [Online]. Available: https://gpac.wp.mines-telecom.fr/

[41] M. Zorrilla, A. Martin, I. Tamayo, S. O'Halpin, and D. Hazael-Massieux, "Reaching devices around an HbbTV television," in Proc. IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (SBMSB), Beijing, China, 2014, pp. 1-7.

[42] C. Howson, E. Gautier, P. Gilberton, A. Laurent, and Y. Legallais, "Second screen TV synchronization," in Proc. IEEE International Conference on Consumer Electronics - Berlin (ICCE-Berlin), Berlin, Germany, 2011, pp. 361-365.

[43] C. Ziegler, "Second screen for HbbTV - Automatic application launch and app-to-app communication enabling novel TV programme related second-screen scenarios," in Proc. IEEE International Conference on Consumer Electronics - Berlin (ICCE-Berlin), Berlin, Germany, 2013, pp. 1-5.

[44] M. Zorrilla, N. Borch, F. Daoust, A. Erk, J. Flórez, and A. Lafuente, "A Web-based distributed architecture for multi-device adaptation in media applications," *Personal and Ubiquitous Computing*, vol. 19, no. 5-6, pp. 803-820, Jun. 2015.

[45] A. Domínguez, M. Agirre, J. Flórez, A. Lafuente, I. Tamayo, and M. Zorrilla, "Deployment of a Hybrid Broadcast-Internet Multi-Device Service for a Live TV Programme," *IEEE Transactions on Broadcasting*, in press (2017).

[46] L. Bassbouss, G. Güçlü, and S. Steglich, "Towards a wake-up and synchronization mechanism for Multiscreen applications using iBeacon," in Proc. International Conference on Signal Processing and Multimedia Applications (SIGMAP), Vienna, Austria, 2014, pp. 67-72.

[47] C. Ziegler, C. Keimel, R. Ramdhany, and V. Vinayagamoorthy, "On Time or Not on Time: A User Study on Delays in a Synchronized Companion-Screen Experience," in Proc. ACM TVX 2017, Hilversum, The Netherlands, 2017, pp. 105-114.

[48] Real Time Streaming Protocol (RTSP), IETF Internet Standard, RFC 2326, 1998, [Online]. Available: https://www.ietf.org/rfc/rfc2326.txt

[49] D. Gómez, F. Boronat, M. Montagud, and C. Luzón, "End-to-end DASH platform including a network-based and client-based adaptive quality switching module," in Proc. International Conference on Multimedia Systems (MMSys '16), Klagenfurt, Austria, Article 38, 4 pages.

[50] M. Montagud, F. Boronat, and P. Cesar. (2014, June). A customizable open-source framework for measuring and equalizing e2e delays in shared video watching. Presented at ACM TVX 2014, Newcastle (UK), [Online]. Available: http://bit.ly/2zzA08N

[51] M. Montagud, and F. Boronat, "On the Use of Adaptive Media Playout for Inter-Destination Synchronization," *IEEE Communications Letters*, vol. 15, no. 8, pp. 863-865, Aug. 2011.

[52] M. Montagud, F. Boronat, B. Roig, and A. Sapena, "How to perform AMP? Cubic adjustments for improving the QoE," *Computer Communications*, vol. 103, no. 1, pp. 61-73, May 2017.

[53] R. Steinmetz, "Human perception of jitter and media synchronization," *IEEE J. Sel. Areas Commun*., vol. 14, no. 1, pp. 61-72, 1996.

[54] M. Montagud, F. Boronat, and H. Stokking, "Early Event-Driven (EED) RTCP Feedback for Rapid IDMS," in Proc. of the 21st ACM International Conference on Multimedia (MM '13), Barcelona, Spain, pp. 323-332.