Synchronized Delivery of Multimedia Content over Uncoordinated Broadcast Broadband Networks

Cyril Concolato   Stéphane Thomas   Romain Bouqueau   Jean Le Feuvre

Telecom ParisTech; Institut Telecom; CNRS LTCI
46, rue Barrault
75634 PARIS CEDEX 13
{concolato, thomas, bouqueau, lefeuvre}@telecom-paristech.fr

ABSTRACT

Existing broadcast networks can deliver identical content to a large number of users. Broadband networks can deliver personalized content to specific users. Hybrid delivery tries to use the best of both types of networks to provide customized services to many users. This paper studies the theoretical aspects and challenges behind the delivery of multimedia content over hybrid broadcast broadband networks, including bootstrapping, synchronization and resynchronization. It presents a solution based on the use of a global clock which does not require communication across networks and is compatible with existing technologies. This solution is implemented in a multimedia player and evaluated against two real-world scenarios, mixing DVB or FM broadcast networks with a broadband IP network. The results confirm the theoretical approach and show that some fine tuning in the networks is needed for a tighter synchronization.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General – *Data communications;* C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design – *Distributed Networks.*

General Terms

Algorithms, Experimentation.

Keywords

Broadcast, Broadband, Hybrid Delivery, Multimedia, Streaming, Synchronization.

1. INTRODUCTION

Modern multimedia devices are capable of connecting to several networks concurrently. Examples of such devices are connected TVs with Digital Video Broadcast (DVB) and Internet capabilities, mobile phones with 3G, WiFi and FM receiving capabilities. This multiple connectivity opens new possibilities for multimedia applications, content providers and network operators. New standardization activities have even started in this area, for example in the Modern Media Transport (MMT) work within MPEG. Among the new possibilities, the hybrid delivery of multimedia content, i.e. the simultaneous use of several heterogeneous networks, raises some interesting challenges. For instance, some applications can rely on broadcast networks and on their good land coverage (e.g. DVB-T or DVB-S) to provide a basic service and on additional IP networks to provide enhancement layers to this service. Alternatively, given the saturation of 3G/4G networks, operators can leverage existing broadcast channels to avoid delivering some content to their customers via the saturated network. Such hybrid delivery of the content can even be used when combining analog and digital content. For example, in case of visual and interactive radio applications as illustrated in [2], the audio content, which represents the major part of the bit rate, can be reused from the analog audio signal of the FM channel.

In this paper, we investigate two scenarios. The first scenario features an audio stream received over a broadcast (analog) channel such as FM radio channels and a visual stream such as an MPEG-4 BIFS stream received over an IP unicast channel. In the second scenario, we consider an audio/video stream delivered over traditional digital broadcast channels enhanced with an additional media stream delivered over an IP unicast channel.

To perform this investigation, we first study the concept of hybrid delivery in general giving some definitions and architecture. We then focus on the specific case of hybrid broadcast and broadband delivery and in particular on the following questions:

- Content matching: how can a client identify and retrieve the different streams from different networks?
- Synchronization: how can a client insure a synchronized playback of the different streams, given the characteristics of each network?
- Re-synchronization techniques: how can a client react when the network conditions change and affect synchronization?

We try to determine, from a theoretical point of view, if a solution for the synchronized playback can be found without requiring modifications to existing broadcast systems and without enslaving one broadcast or broadband equipment to the other. Then we confront the theoretical results with practical broadcast networks for T-DMB (Terrestrial Digital Multimedia Broadcasting) and DVB delivery, based on the use of the available timing information, such as the DVB TDT (Time And Date Table from MPEG2-TS) and RDS (Radio Data System) CT (Clock Time) field.

The rest of this paper is organized as follows. Section 2 presents related works in this area. Section 3 gives some definitions and proposes a description of the problems related to the hybrid delivery of multimedia content. Section 4 presents the methods we used to achieve a synchronized presentation in the given scenarios. Section 5 presents the implementation aspects. Finally, Section 6 concludes this paper and proposes future work.

MMSys'12, February 22-24, 2012, Chapel Hill, North Carolina, USA.

2. RELATED WORKS

There exist multiple previous works in the area of multi-network content delivery. In [4], Boronat and al. make an inventory of existing distributed media presentation proposals as of May 2008, with a focus on synchronization issues. They distinguish continuous and event-based synchronizations as well as live and synthetic (retrieval-based systems such as Video on Demand) contents. In our work, we rely on this analysis but with a focus on the coupling of broadcast and broadband networks, with the use of existing delivery formats or protocols such as MPEG-2 TS or MPEG Dynamic Adaptive Streaming over HTTP (DASH).

In [9], Aoki et al. also envisaged the hybrid broadband broadcast delivery of audiovisual contents, including the use of MPEG-2 TS. They report the problem that current broadcast and broadband encoders do not share their system time clocks. To overcome this limitation, they propose to unify the broadcast and broadband delivery by using IP on both and by using the same media time stamps on both. In this work, we try to rely as much as possible on existing technologies and to keep both worlds independent.

The work of Biersack and Geyer [3] is related to our work as they study the synchronization issues of distributed streams. However, the authors assume a return channel, used to exchange control messages, in a manner similar to the RTCP protocol. In our work, we do not rely on the use of the return channel because in our broadcast environment, it may not be available.

In [7], the authors propose an algorithm to synchronize two video files played by two different clients linked with a bi-directional communication based on the hash code of the video. In our work, we do not consider synchronization across clients but rather the synchronization of streams within a client.

In [6], Prarthana et al. defined a way to synchronize multiple videos of the same scene based on the audio track. This approach is interesting for scenarios where some redundancy (e.g. fingerprints) is added to a stream to help the client to synchronize it with other streams. In our work, because of bandwidth limitation, we consider this approach not practical. We rely only on timing information, and as much as possible on existing standard information, to achieve synchronization.

More recently, Evensen et al. [1] studied the use of a hybrid delivery of a single media stream using different networks. They investigated how to segment and schedule the segments of the media stream over all the available networks to guarantee that the playback will be possible, with as few interruptions as possible. Our work is different in the sense that a unique network is used to deliver each stream.

As far as we know, there is no previous work focusing on the synchronization, within a client and without return channel, of multiple streams coming from a broadcast network and additional broadband networks with current broadcast systems.

3. HYBRID DELIVERY SYSTEMS

This section summarizes the three main issues that we have identified in the delivery of multimedia content over multiple heterogeneous networks. Our work addresses issues from a receiver perspective. We start first with some definitions and an architecture proposal to clarify the scope of our study. We then describe the issues of stream matching, stream synchronization and how to handle a loss of synchronization.

3.1 Definitions

In this paper, we consider the delivery of M media streams (m_i, 1 <= i <= M). Each stream can be an elementary stream, such as an audio stream, or a multiplexed stream, such as an MPEG-2 Transport Stream or a stream of DASH segments based on the MP4 file format, or a file on a Web server which may be intended to play at a given time.

We also consider that these streams are delivered over N networks (n_j, 1 <= j <= N). A network is defined as a set of means to deliver a media stream from the server to the client. These means are made of protocols such as the Real-Time Protocol (RTP) or HTTP and physical mediums such as wired or wireless links. The characteristics of such networks are:

- the throughput,
- the loss ratio,
- the latency, i.e. the difference between the time at which an element of the media stream is sent by the server and the time at which it is received by the client,
- the maximum jitter, i.e. the maximum variation of the latency.

We consider that two networks are different if their characteristics are different, even if the protocols or physical mediums are the same. Finally, we consider that the M streams are delivered over the N networks using S servers (s_k, 1<= k <= S).

In the context of this paper, we will consider that the delivery of the multimedia content is hybrid if S > 1 or if N > 1. Indeed, if S=N=1, all media streams are delivered under the same conditions, so the hybrid term does not apply and synchronization issues are out of the scope of this paper. The delivery can be hybrid if N=1 and S>1, for instance, when two independent servers provide two streams with the same network characteristics. This still presents some challenges to the client to present the data synchronously, especially when the initial requests are not simultaneous. In this paper, we will consider that M>1 even if, as described in [1], hybrid delivery strategies can also be envisaged with M=1.

3.2 Architecture for the hybrid delivery

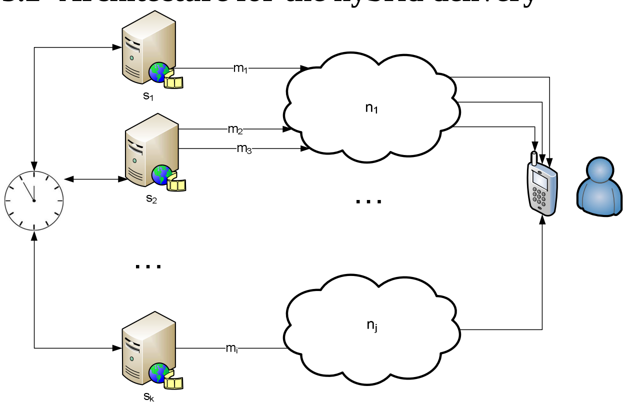

Figure 1 - Overall architecture of the hybrid delivery of multimedia streams

As illustrated in Figure 1, in our work, we assume that all servers are independent with respect to the coding of their media streams (with the possible exception of scalable streams). In particular, the timestamps of media units produced by different servers are not related, e.g. they can start with some different values. However, we also assume that each server has a mean to obtain a common clock value (e.g. using GPS, using the Network Time Protocol (NTP)) and that this clock value is carried usually in-band, within the media stream.

As described in [4], there are various delays that can affect the delivery of a single media stream. Some of the delays are due to the server processing, the network protocols and layers and some to the player processing. In hybrid delivery situations, the delay and jitter of a stream relative to other streams are particularly relevant.

3.3 Bootstrapping

In typical multimedia streaming approaches, an initial bootstrap phase is needed to locate the streams delivered using a given protocol. This phase uses either some out-of-band or in-band bootstrapping information. For instance, in RTP streaming, an SDP (Session Description Protocol) description can be used. Another example is the Program Map Table (PMT) in the MPEG-2 TS format. The hybrid delivery of multimedia streams as defined in the previous section follows the same approach but with new requirements. In particular, the bootstrapping information should enable the player to locate multiple resources from different networks using different location schemes. In that sense, the PMT is not satisfactory. Moreover, the bootstrapping should provide the player with a description of slave/master relationships (if any) [4].

There are several potential formats to express the bootstrapping information. For instance, the SDP format could be extended to such intent for the slave/master relationship. The scene description formats such as SVG, HTML 5 or SMIL are already able to reference media streams coming from different locations. The HTML 5 draft uses a *mediagroup* attribute to indicate to the browser that different videos shall be synchronized, but without additional information on master/slave or on synchronization tolerance. The SVG and SMIL languages propose more information in this respect with for example the *syncMaster* attribute or the *syncTolerance* attribute. Finally, we can also cite the MPEG DASH MPD (Media Presentation Description) language which allows indicating multiple media representations from different servers, with default hard synchronization. However, the format is currently limited to HTTP URLs only.

3.4 Stream Synchronization

3.4.1 Clock synchronization

We are interested here in inter-stream synchronization. When the streams are generated by the same source, inter-stream synchronization can be achieved with the insertion of packets inside each stream with the same clock information in it. Each video or audio packet is then associated with a value of this clock. In MPEG-2 TS such a clock is called PCR (Program Clock Reference). It is interesting to point out that the seed PCR value refers to a precise moment in the media but has no signification outside the media. Therefore this inter-stream synchronization approach cannot be used if the streams are delivered by independent servers.

In order to achieve inter-stream synchronization, several other options are also possible. For instance, the HbbTV standard [11], which deals with some hybrid content from broadcast and from broadband networks, relies on events to synchronize the broadcast media data (audio/video) with the broadband HTML/JavaScript data. The synchronization here is done with a specific message inserted inside the broadcast stream. In the current HbbTV specification there is no timing information sent with this message. The synchronization is done on a best effort basis as it highly depends on the time needed to interpret the message by the application. If one adds timing information inside the message packet, this problem can be avoided: accurate synchronization between a continuous and a discrete media can be achieved. In DVB, this can be done by filling the *eventNPT* field of a *StreamEvent*. Unfortunately, this is not used in the HbbTV standard, as it would require additional stream-event information for each frame.

Another option for the hybrid synchronization is to use a clock shared by every stream, but independent of the coding clock generated by each server. Each packet needs to be associated with a value of this shared clock. In this work, we used a global clock based on the UTC time. As usual with clock dependencies, one has to cope with offset problems (also called insertion error, due to a difference between the UTC time available to the encoder and the absolute Media Time indicated within the stream) and drift problems (the clocks don't run at the exact same speed). Figure 2 illustrates both issues. With the approach described in this paragraph, a precise synchronization can theoretically be achieved when both the offset and the drift are null. The rest of this paper will be based on this later approach.

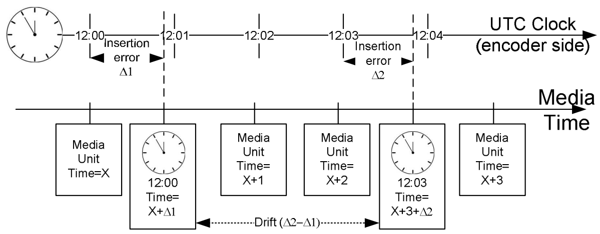

Figure 2 - Media Time and UTC clock: insertion error and clock drift

3.4.2 Playback procedure

Let us assume the bootstrapping phase provides the location of all the streams at the same time (i.e. one stream is not used to provide the location of the others). Because of the latency of each network, it is unlikely that media units from all streams associated with the same UTC time will arrive at the same time. Also the mapping between the stream timing and the hybrid timing may not be known at that time. However the client is expected to start the playback as fast as possible to achieve the best quality of experience for the user. Therefore, some complex playback procedure might be needed.

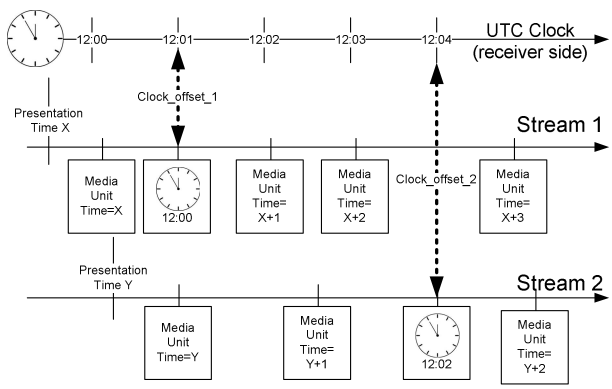

Figure 3 – Clock relationships for a 2-stream hybrid content

Let us assume the client plays a 2-stream hybrid content as shown in Figure 3. Stream 1 will deliver a first global UTC clock value after some time. At that time, the client computes clock_offset_1. Similarly, the client can compute clock_offset_2. Then, the client has several choices:

- If clock_offset_1=clock_offset_2, both streams are already synchronized, it can start or continue the playback;
- If clock_offset_1>clock_offset_2, the content of stream 2 arrives before the content of Stream 1. Depending on several factors (bootstrapping information, delivery protocols, playback policy…), the player can decide to pause and buffer stream 2 and wait for stream 1 data to arrive. It can also decide to fetch data from stream 1 in the future. For example, with the RTSP protocol, a PLAY request with a range attribute in the future can be used. Using the DASH approach, if the content has been produced with some latency compared to live, and if the segments are not too large, the player may request segments in the future.

3.5 Stream Re-Synchronization

Re-synchronization happens when the synchronization of a stream is lost despite basic or preventive control techniques as defined in [4]. For instance, the buffer associated to a stream may underflow in the following cases: if network conditions have worsened; or if the latency was underestimated. In such a case, the client may:

1. Keep playing the streams which have sufficient buffered data. The late packets for the underflowed stream may be:
    a. Played late, losing synchronization for some time, potentially adjusting the playback speed when possible (e.g. increasing/reducing audio silence duration) to recover;
    b. Skipped, in the hope that future packets will arrive before their playback deadlines.
2. Pause the playback and wait for the underflowed stream buffer to be sufficiently filled to resume.

In all cases, the player may adapt its buffer requirements or try to prefetch data. Such behavior can be decided at the player's convenience or according to the information made available at the bootstrap.

4. PROPOSED SOLUTIONS

4.1 General considerations

As explained in 3.4.1, for all scenarios, we rely on the existence of an association of each packet of each stream with a global UTC time. In the context of MPEG-2 TS, the DVB TDT (Time and Date Table) construct already enables the carriage of a UTC clock. It is a good candidate for enabling hybrid synchronization, but has several problems:

- The time provided in the TDT is not related to specific video or audio frames. It is in general used to provide end-user timing for electronic program guides;
- TDT packets are not sent very often;
- The TDT time expresses a time in seconds and its accuracy is therefore highly dependent on its insertion into the MPEG-2 TS, as described in Figure 2. For instance, for a typical 24.8 Mbps[1] transport stream, the insertion of the TS packet carrying the TDT (Δ) can be adjusted by steps of approximately 60 µs.
- The TDT time may drift compared to the PCR clock as shown also on Figure 2.

The rest of this section will evaluate these problems on the specific scenarios.

4.2 Synchronizing broadcast FM audio and IP MPEG-2 TS visual content

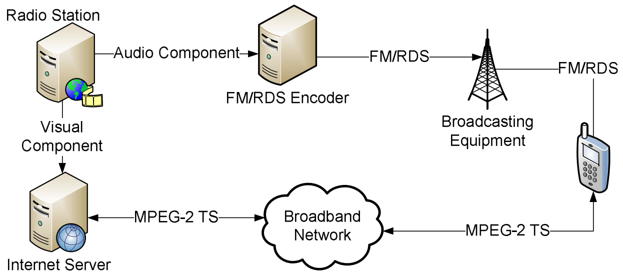

Figure 4 - Architecture of a hybrid delivery system for audio FM and visual IP content

For this scenario, we use the system described in Figure 4. The radio station produces the audio and visual contents together, synchronously. The audio content is transmitted to an FM/RDS encoder which encodes and modulates the audio signal and the Radio Data System (RDS)[2] data. The visual content is delivered by an Internet server (HTTP, Icecast, or RTP) in the form of an MPEG-2 TS stream, for instance according to the T-DMB standard.

In this system, we can rely on the ability of the audio channel to transport the bootstrapping information. Indeed such information can be transported for example within a RDS structure called ODA. These data are sent periodically at a low bit rate (typically a few bytes every second). However, since the FM transmission is prone to errors, the location of the secondary streams is only known after some, possibly long, period of time. Therefore, the bootstrapping phase is as follows. The audio channel is setup first, and only after some time, the second stream can be fetched and the synchronization can be achieved, as described in 3.4.2. However, one can imagine that the location of the stream will not change very often and in most cases, the bootstrap information could be cached.

In terms of synchronization, the audio channel periodically transports RDS data. Within this data, a UTC Clock Time field (CT) transmits the time at which the RDS data was encoded. In our system, this UTC clock value is compared to the TDT clock value inserted in the MPEG-2 TS visual content. The RDS clock has the following characteristics which may impact synchronization. In an error-free environment, the RDS standard specifies the CT field is retrieved approximately every minute with 100 ms accuracy. Additionally, the RDS clock is computed and inserted in the FM stream either in the radio station (at the same moment as the audio content is generated) or in the broadcasting antenna. This latter case introduces an offset

[1] http://www.digitalbitrate.com

[2] http://en.wikipedia.org/wiki/Radio_Data_System

between the actual audio content generation time and its associated clock value. This offset corresponds to the delay to deliver the audio between the station and the antenna. It is supposed to be constant over time. For the French main territory, such delay can reach a value as high as 750 ms and have a great impact on the hybrid synchronization as we described in 3.4.1. In this work, we assume that such delay can be mitigated by a prior calibration of the network by the operator.

In total, this means that hybrid synchronized playback cannot occur until: **a)** the location of the MPEG-2 TS is retrieved (from no time if cached, to some minutes, if not cached); **b)** the RDS time is retrieved (>1 min); **c)** the MPEG-2 TS TDT time is retrieved; **d)** enough MPEG-2 TS data is retrieved to fill the buffer and mitigate the jitter on the IP network.

In order to evaluate c) and the associated drift, we have measured the insertion of the TDT in a live DMB stream produced by a commercial encoder. The results are shown in red in Figure 5 for a 5-hour-long trace and shows that the inter-arrival time of TDT packets (expressed in PCR time) is roughly constant, around 1 second. Therefore, c) can be neglected compared to b).

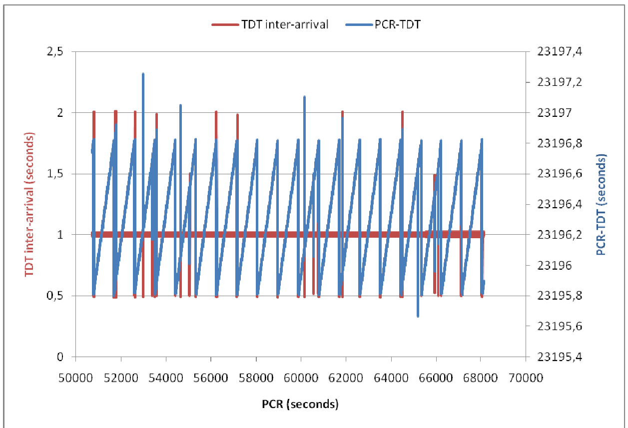

Figure 5 – TDT/PCR drift in commercial T-DMB content

Figure 5 also shows in blue that the difference between the time carried in a TDT packet and the PCR value associated to this packet suffers from a drift, roughly a second (shown as the right ordinate values between 23195.8 and 23196.8) but gets however corrected every 1000 PCR seconds as displayed in abscissa. Therefore, if we add the mis-accuracy of the RDS time (100 ms), the uncertainty of the insertion of the TDT c) which is neglected compared to b) (0 ms) and the drift of the TDT with respect to the PCR (1000 ms), we might end up with more than 1.1 second of synchronization error. Such value cannot enable to achieve any augmented scenario (audio dub, hard-synchronization interactivity), which usually require less than 100-150 ms [10], but is enough for many interactive services. However, if precise TDT insertion was achieved, we could reach 100 ms synchronization, which would be sufficient for most applications.

4.3 Synchronizing broadcast MPEG-2 TS and IP MPEG-2 TS content

For this scenario, we use the system described in Figure 6. The top part of the figure represents a typical production/broadcast chain for digital television contents. The bootstrap for this scenario relies on the MPEG DASH MPD.

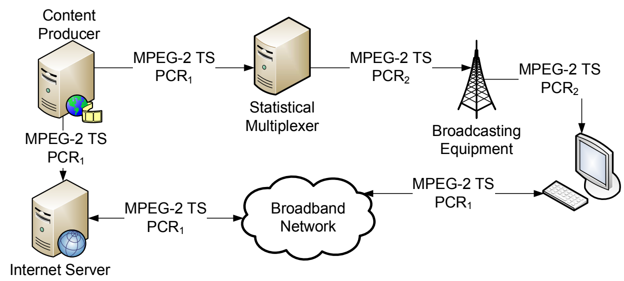

Figure 6 - Architecture of a hybrid broadcast MPEG-2 TS and broadband content

A content producer produces a (single program) MPEG-2 TS content that is delivered to two entities. The main audio/video MPEG-2 TS content is delivered to a multiplexer connected to broadcasting equipments (terrestrial network, satellite …). Some additional content, such as an extra view for stereoscopic services, is delivered to an Internet server. The multiplexer will typically decode, encode and multiplex the different sources into an MPEG-2 TS content, and the PCR timing information will be rewritten in the process. Hence, even if the two streams created by the content producer share common PCR values initially, the client will receive data which do not have the same timing.

Therefore, in such a system, we propose to use the same TDT-based approach (as in 4.2) when the IP content is based on MPEG-2 TS. Similar solutions can be achieved with MP4 based content (DASH) by using the NTP timestamp carried within DASH segments (ProducerReferenceTime boxes). RTP delivery can also use the NTP timing. As explained for the previous scenario, the TDT-based approach relies on the precise insertion of the TDT here in both MPEG-2 TS. We measured it on real DVB-T content. The results are shown in Figure 7. It shows that the inter-arrival between two TDT is rather constant but long (25 seconds). We also see that the difference between the time expressed in the TDT and the PCR time of the TDT packet is not constant. The difference varies within a 2 seconds range. This shows that in a scenario of hybrid delivery of DVB-T based MPEG-2 TS content on both broadband and broadcast networks, we could have a mis-synchronization in the order of 4 seconds.

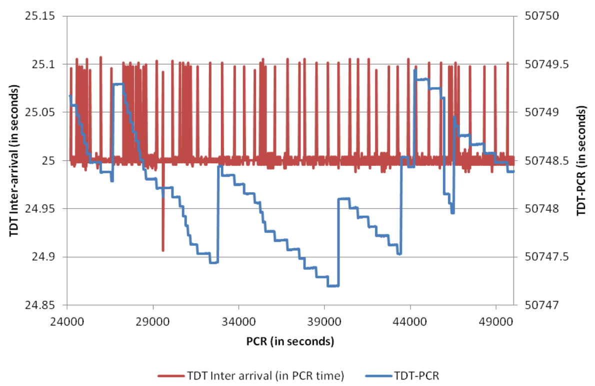

Figure 7 - TDT/PCR drift in real-world DVB-T multiplex

5. IMPLEMENTATION

To validate the theoretical aspects of the ideas proposed in 4, we created a test sequence and modified the implementation of the GPAC player [2] as described in this section.

5.1 Test sequences

As a basis for both scenarios detailed in 4.2 and 4.3, we used a sequence similar to the bipbop Apple sequence[3], but for PAL configuration with 25 fps and 1 second interval between beeps.

For the first scenario, we filtered the audio elementary stream out of the MPEG-2 TS segments, and kept only the video stream. For the FM content, we generate a raw audio stream, locally on the client, which contains an audio beep every second to match the bipbop video timer.

For the second scenario, we used the bipbop video-only MPEG-2 TS and the bipbop audio-only MPEG-2 TS located at two different HTTP servers. We used a PCR in the two TS streams and added a random offset to all PCR/PTS/DTS values on one of them.

5.2 Player implementation

In order to play a hybrid delivered content, we had to modify the GPAC player in several ways.

First, we added a new module to implement the bootstrapping method. This module is in charge of analyzing on which network the request shall be made and to send the address to the downloader module corresponding to the appropriate network. We also added the possibility to the GPAC player to have a download entity per network (or more exactly per group in the DASH MPD).

Then, we had to modify the player to handle synchronization issues. In most common MPEG-2 TS player, the display of every audio or video frame is synchronized with the PCR clock carried inside the multiplex. As said previously, in case of two hybrid delivered MPEG-2 TS, the clock of every media is also enslaved to a common TDT clock. We implemented a mapping of PCR and TDT using a method of the player already used with the RTP/RTSP/RTCP protocol. Indeed, the use of random offsets for timestamps in RTP packets forces the player to remap the timestamps to the media time. This is done once RTCP or RTSP messages providing correspondence between the first timestamps and the media time are received. The only difference lies in the fact that in RTP, one can buffer or skip the packets until the first RTSP/RTCP message is received. In the FM/TS or TS/TS case, the player cannot buffer for such a long time (up to 1 min using the FM Clock Time clock) until all inter-stream clocks are received. It either has to drop the packets until all mappings are received on all streams, or it can play some stream (e.g. FM only) until the mapping of time is known on other streams. This case shows the importance of knowing which stream, if any, is the master.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we defined the concept of hybrid delivery. We presented the different related issues: bootstrapping, synchronization and resynchronization. We proposed some solutions for the synchronized playback of hybrid delivered streams in two different scenarios, mixing broadcast and broadband delivery, without return channel and with a focus on the client behavior. These solutions are based on the use of a global UTC time carried in-band in all networks. We presented the modifications made to the GPAC player to implement these solutions. We showed that theoretically it enables the synchronized playback with practical usages, but that in real networks, it highly depends on the care with which the UTC time is inserted in the stream. In future works, we plan to evaluate the consequences of the modifications of the different network conditions (latency, jitter). The implementation of a module for the FM/RDS reception on the Samsung's Galaxy S2 phone is ongoing and should provide more concrete results on real FM networks.

7. ACKNOWLEDGMENTS

This work was partially funded by the French Research Agency in the course of HybRadio project.

8. REFERENCES

[1] Kristian Evensen, Dominik Kaspar, Carsten Griwodz, Pål Halvorsen, Audun Hansen, and Paal Engelstad. 2011. Improving the performance of quality-adaptive video streaming over multiple heterogeneous access networks. In Proceedings of the second annual ACM conference on Multimedia systems (MMSys '11). ACM, New York, NY, USA, 57-68.

[2] Cyril Concolato, Jean Le Feuvre, and Romain Bouqueau. 2011. Usages of DASH for rich media services. In Proceedings of the second annual ACM conference on Multimedia systems (MMSys '11). ACM, New York, NY, USA, 265-270.

[3] E. Biersack and W. Geyer. Synchronized delivery and playout of distributed stored multimedia streams. Multimedia Syst. J., 7(1):70–90, 1999.

[4] Fernando Boronat, Jaime Lloret, and Miguel Garcia. 2009. Multimedia group and inter-stream synchronization techniques: A comparative study. Inf. Syst. 34, 1 (March 2009), 108-131.

[5] Hao Zhang, Chuohao Yeo, Kannan Ramchandran. 2008. VSYNC – A Novel Video File Synchronization Protocol. MM '08 Proceeding of the 16th ACM international conference on Multimedia

[6] Prarthana Shrstha, Mauro Barbieri, Hans Weda. 2007. MM'07 Proceeding of the 15th ACM international conference on Multimedia

[7] Jean Le Feuvre, Cyril Concolato, and Jean-Claude Moissinac. 2007. GPAC: open source multimedia framework. In Proceedings of the 15th international conference on Multimedia (MULTIMEDIA '07). ACM, New York, NY, USA, 1009-1012.

[8] Thomas Stockhammer. 2011. Dynamic adaptive streaming over HTTP: standards and design principles. In Proceedings of the second annual ACM conference on Multimedia systems (MMSys '11). ACM, New York, NY, USA, 133-144.

[9] Aoki, S.; Aoki, K.; Hamada, H.; Kanatsugu, Y.; Yamamoto, M.; Aizawa, K.; , "A new transport scheme for hybrid delivery of content over broadcast and broadband," Broadband Multimedia Systems and Broadcasting (BMSB), 2011 IEEE International Symposium on , vol., no., pp.1-6, 8-10 June 2011

[10] Steinmez R Human Perception of Jitter and Media synchronisation, IEEE Journal on Selected Areas in Communications, 14(1), January 1996, 61-72.

[11] HBBTV, ETSI Technical Specification 102 796, "Hybrid Broadcast Broadband TV", V1.1.1 (2010-06)

[3] http://devimages.apple.com/iphone/samples/bipbopall.html