SERVER-SIDE CLIENT SYNCHRONIZATION FOR WATCH TOGETHER APPLICATIONS USING CMAF LOW LATENCY

P. Gendron

Harmonic, France

ABSTRACT

Watch Together is an application that has been widely deployed during the COVID-19 global health crisis. Early results show a much higher viewing time when the feature is activated. The synchronization between the A/V streamed content, combined with the need to have a low, end-to-end latency compatible with the user interactions through integrated social media apps, is challenging, especially when Watch Together is deployed on all devices.

The classical approach involves delaying or accelerating each client and reaching an equilibrium point after a certain time. The risk with this approach is running into an overflow or underflow situation. Since it is not a standard solution, it requires adaptation on each client, which has a different ABR behavior. This results in a lot of effort with unpredictable outcomes in some cases.

This paper will propose a scheme that uses the built-in synchronization provided in CMAF low latency for both DASH and HLS, enabling a sub-second time delay between clients based in the same geography.

INTRODUCTION

Sharing with friends is the ultimate experience during live sports events. Whether you are in stadium or in a bar, feeling like you are part of a community is essential. In 2020, the COVID-19 pandemic hindered socialization, causing a rise in emerging applications aimed at virtually connecting groups of people attending popular sports events. Watch Together applications have been branded under different names by several operators and widely deployed for live and SVOD services. Early results show a much higher viewing time when the feature is activated. That’s because having the ability to watch sports in social settings combined with the capability to post stories on social media increases the enjoyment of watching live events. See (1) “Co-Watching: Creating the Power of Togetherness,” https://www.sportsilab.com/cowatch-ppi for more details on the attractiveness of Watch Together applications.

Creating Watch Together services can be technically challenging, as there needs to be synchronization between the A/V streamed content, combined with a low, end-to-end latency compatible with the user interactions through integrated social media apps, especially when deployed on all devices.

The classical approach involves delaying or accelerating each client and reaching an equilibrium point after a certain time. The risk with this approach is running into an overflow or underflow situation. The first emerging solutions are all proprietary, not relying on standardized approaches, and generally require adaptation on each OTT client, which has a different ABR behavior. This results in a significant effort to integrate the solution into the various client platforms and players that the service provider is targeting, with unpredictable outcomes in some cases. When more than a few clients need to interact, guaranteeing full synchronization is challenging. The synchronization function will require dedicated servers that will financially impact service profitability.

This paper will first describe the concept and building blocks of a generic Watch Together application. Then it will describe the state of the art for currently deployed services and conclude with a proposed scheme that uses the built-in synchronization provided in CMAF (2) low latency for both DASH (3) and HLS (4), enabling a sub-second time delay between clients based in the same geography. The results are based on testing performed on various CMAF low-latency clients in both HLS and DASH modes.

WATCH TOGETHER CONCEPT AND GENERIC ARCHITECTURE

Watch Together Application Concept

While Watch Together services may have different commercial names and different implementations for live content and VOD assets, they generally have a few similarities:

- A group of people wants to see a program as if they were co-located (or as if they were really in the stadium for a sports event) and socialize about it while they are watching
- This can apply to live events and pre-recorded content available on the service provider's back-end (i.e., typically a VOD asset)
- The viewing is possible on a wide range of devices and not limited to a brand or a streaming protocol. Viewing can be done from any location where the streaming service is available (we will assume the location is in the same country).
- As socializing is an important part of the experience, the Watch Together application should not only provide the program video content but also offer viewers an easy way to interact with the group. This can be a simple message chat box or something smarter with the capability to have multi-user video chat.
- The construction of the watching group should be easy and dynamic, allowing last-minute scheduling and the addition of new participants even during the event. Users should be able to start a watching session and invite friends to join in using a link distributed by the organizer.
- To ensure a smooth experience and a way to socialize in real time, as if the users were co-located, the system should ensure that all the players from the different users are aligned in time. Frame accuracy is not required but the inter-device delay should be kept under one to two seconds.

The two categories of content that can be watched together have some requirement differences:

- For VOD content, the delivery latency is not critical (startup time is much more critical for a better user experience) so the synchronization between players is the only dimension to consider.
- For live events, it is well understood that sports are the main category of events driving the needs of Watch Together experiences. For sports, it is well known that the viewing experience can be ruined by the lower latency of social networks or concurrent delivery path of cable services. Therefore, OTT video streaming services are now moving to low-latency delivery as the major streaming protocols, HLS and DASH, both offer a low-latency extension in their protocol. The consequence is that the synchronization between the different players shall be within the same one-to-two second range but should also be done with a short end-to-end latency.

Watch Together Applications Architecture

Although they can be built with different approaches, the Watch Together applications need to have various building blocks to offer the service, as shown in Figure 1.

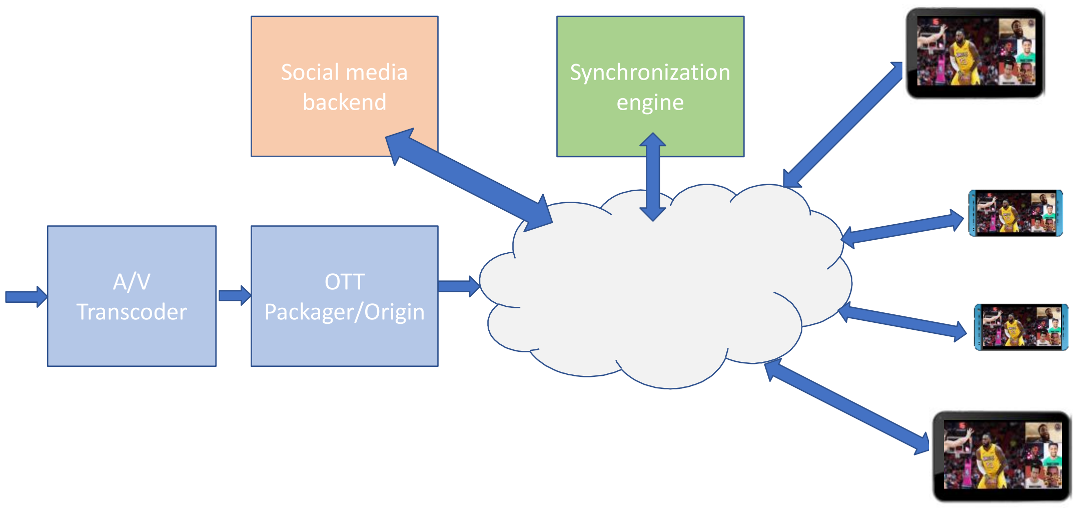

Figure 1 - Watch together high-level architecture

Video Channel Preparation

The A/V transcoder and OTT packager/origin building blocks are regular OTT service elements. They should make available the content to the players using the popular streaming formats, HLS and DASH. There is nothing specific in this part of the workflow except that for sports applications, low-latency delivery enables a better user experience. This point will be covered in the section related to CMAF-based delivery.

Social Media Backend

As mentioned before, socializing with friends while watching a sports event, a TV show or a series is an activity that's in high demand. This covers mostly SMS, the chat and video chat applications. As one can expect, these demands are greater with younger audiences.

Therefore, the Watch Together application should have or be connected to a social network architecture. As this is usually linked to the service CMS, the backend can be dedicated to the Watch Together application.

Synchronization Engine

As reported in (1), *"anyone who has ever tried to watch a game with friends or family using FaceTime or Skype knows that video synchronization is the biggest technical challenge to a co-watching experience*."

The Watch Together architecture should therefore have a mechanism that ensures all users are on the same point in the media timeline. The next sections of the paper will develop the root causes why this is not intrinsic to OTT content distribution, as opposed to a broadcast distribution. This synchronization engine should be reliable enough to support the diversity of networking conditions affecting the different users of the group, keeping in mind that these conditions can evolve during the session (e.g., some users can be on wireless home access or even on mobile networks). The synchronization should have mechanism to dynamically adapt to any drift a player can be exposed to.

A 2018 report said that 63% of sports fans were reluctant to renew subscriptions to a streaming platform because of buffering and quality issues. This shows that the networks are not yet perfect and adding a co-watching experience on top of classical video distribution requires even better end-to-end control over the delivery.

Player Application

The player application installed on the device should manage two main flows:

- The main audio/video stream delivered in HLS or DASH.
- The social media stream that is delivered typically using WebRTC (5) for the video chat part.

The main player part should be driven by either an overlay process controlled by the synchronization engine or from the stream to properly decode and display the content in such a way to ensure synchronization with the other players.

The social media part of the application should provide a state-of-the-art method to communicate instantaneously, with no noticeable delays to enable live group interaction. The expected level of performance is on par with using WhatsApp or any other messaging application, but it is integrated in a single application.

The player may also have a mechanism to properly balance the bandwidth allocation between the video (main channel) stream and the social media feeds, with possibly a degraded mode where in the case of bandwidth limitation the main channel will be privileged and the quality of the social media feed may be reduced or even fall into a degraded mode (i.e., no social video feed, just a text chat).

STATE OF THE ART OF WATCH TOGETHER APPLICATIONS

Existing Services

Several new services have popped up in the last year as the first generation of applications became mature enough for production. Their creation was also driven by the new needs shaped by the pandemic and stay-at-home orders.

Figure 2 is a sample of major services providing a Watch Together feature either for live or VOD on different types of client platforms.

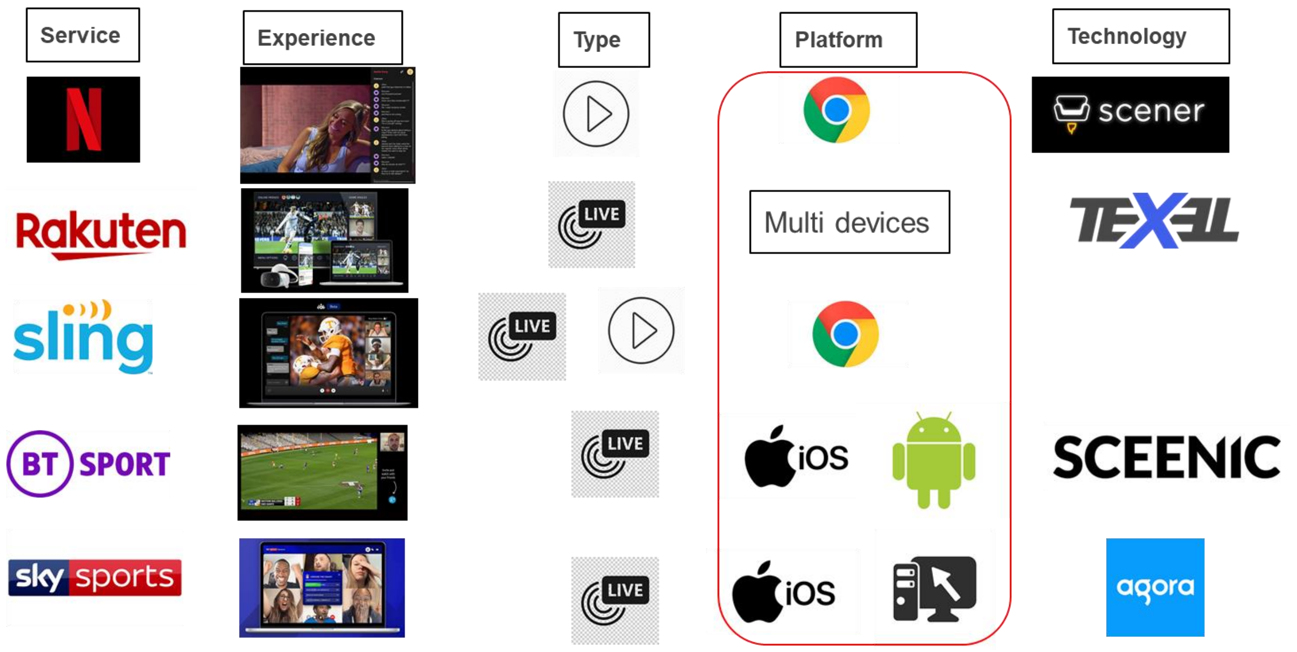

Figure 2 - Major Watch Together services

These offerings rely on different vendor technologies that have been developed. Nevertheless, there are some commonalities between them that will be examined in this section, which will also define the current state of the art.

Watch Together applications target either VOD, live or both. Synchronizing different viewers on a live stream is different from synchronizing everyone on a viewing session for a VOD asset, which is fully available on a server. Therefore, the technology could be different.

The current applications all rely, for the main audio/video feed synchronization, on an overlay synchronization protocol. As a result, there is a need to make a specific, device-dependant integration, at least per device families. This aspect explains that for these first-generation services, a limited set of devices and platforms are offering the services. It is well known that integration of non-standard mechanisms or protocols can be very long, painful and require endless work as new platforms and devices enter the mix over time.

We will now quickly go into the main concepts used for the player synchronization in these first-generation applications.

Current Concepts Used to Synchronize the Players

The first generation of Watch Together applications work with two types of approaches:

- Group locked on a master player
- Group locked on a central point computing a weighted average of delays that each player needs to add to the live edge

Both approaches assume that to synchronize the different users the players need to have a variable playing rate to follow a target by playing faster or slower than nominal speed. This classical approach involves delaying or accelerating each client and reaching an equilibrium point after a certain time. The risk with this approach is running into an overflow or underflow situation. Since it is not a standard solution, it requires adaptation on each client, which has a different ABR behavior. This results in a lot of effort with unpredictable outcomes in some cases. When more than a few clients need to interact, guaranteeing full synchronization is challenging, especially when each client belongs to a different family of players.

The first approach where the group needs to be locked to a master player assumes that this master (usually the first member of the watch party) has a reliable enough connection to be online and reachable to each of the other players at any time. Indeed, the synchronization is made at the startup or when a new user joins the watch party, but it needs to be monitored over time, as any network glitch experienced by any user can affect the synchronization of the group.

The second approach makes use of the synchronization engine located in the cloud or server side to define the nominal live edge all the players should align to. This approach is more reliable in that it doesn’t rely on one of the watch party member players. Rather it requires an overlayed synchronization protocol between this central point and any of the players.

The current deployed services based on one of these approaches shows a synchronization error between the players of about three to four seconds at the worst, and a few hundred milliseconds for the best-case scenario. This means that regardless of the potential additional latency these systems may add to the live edge, and the players can still be up to three to four seconds off or even completely out of sync in the case of network problems, which can be problematic for live sport events.

Many of these applications, not built with the low-latency approach are offering relative synchronization but with a latency up to 30 seconds compared with the live broadcast. This can cause frustration when social networks not attached to this application offer live information much faster.

EMBRACING THE POTENTIAL OF CMAF-BASED LOW-LATENCY STREAMING STANDARDS

This section will describe how CMAF can be utilized to achieve better performance and simplify the construction of a Watch Together system. We will first briefly review the way latency is managed with regular DASH and HLS and see to what extent it clarifies the current Watch Together synchronization situation (state of the art described previously). Then we will describe the elements that can be used in the CMAF low-latency toolbox to achieve natural synchronization between the different players.

Latency Management in Regular DASH and HLS Delivery

OTT streaming is based on two major delivery formats, HLS and DASH, which are characterized by a delivery latency in the range of 10 to 30 seconds. OTT latency is much more than what a traditional broadcast (i.e., cable, satellite or terrestrial) offers (usually in the range to three to eight seconds from acquisition of signal to display). The significant difference in latency is due to there being a totally different way to deliver the content to users. For broadcast applications, a transport stream is sent over the air, and the player needs to tune to the stream to demodulate/decode the signal.

Compared with broadcast, OTT systems make available a set of files that the player should retrieve, creating requests (usually using HTTP protocol over an IP network). To make these requests the players need to be informed of the new resource (i.e., the new segment) availability, download the resource, and then put it in a buffer before starting the decoding process. This difference in approach explains why, combined with smart buffer management (required by delivery over the open internet), it cannot have the same guarantee as delivery over a dedicated, managed network.

As the primary and initial usage of both HLS and DASH was mostly targeting VOD, the latency has not been addressed as a critical parameter, and this was not a topic of differentiation between the players. The result is that most regular OTT services currently deliver quite large latency compared with the live stream, but what's even worse (for an application like Watch Together) is this latency has a lot of variation from one player to another.

Different methods to address the ABR strategy and associated buffer management, identified as the main problem, have been developed as proprietary solutions by player vendors. **This is not under the service provider's control and therefore a variety of latency can be experienced by subscribers depending on the player/platform they use**.

Latency variation can occur in traditional OTT delivery because players may have different strategies to address the start of the session. As shown in Figure 3, a user hits the play button during a segment having several key frames. These different player strategies to start the playback can produce a latency variation in the range of a segment duration (typically six to 10 seconds):

- The player can download “segment i” and start to display the content starting from beginning of this segment
- The player can download “segment i,” parse it and start to display the content beginning from the closest key frame in the past, which compared with the previous approach can provide a latency short by a fraction of a segment duration
- The player can also decide to wait until the new segment is available to start downloading and displaying it

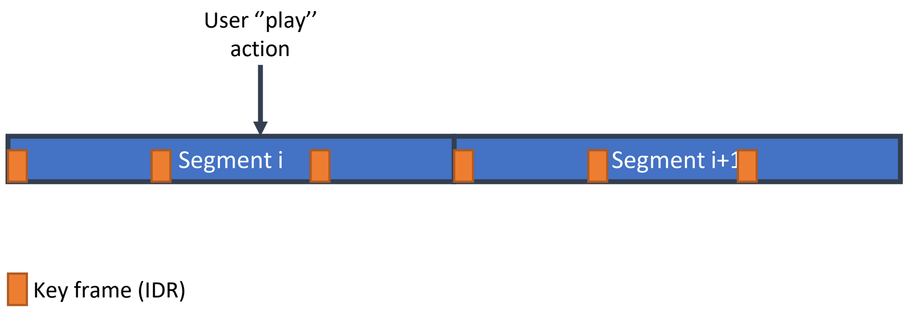

Figure 3 - Session start in regular OTT

The CMAF Low-Latency Tools to Ease Player Synchronization

The CMAF standard was developed to create a common solution for the media files delivered using HLS and DASH, but it also considered the more recent needs to deliver live OTT services with a latency about the same as broadcast. For that purpose, a set of tools and recommended practices have been described to simplify and help the player to master this low-latency requirement.

Compared with some of the distributed synchronization approaches mentioned previously in this paper, the synchronization enabled by using a low-latency CMAF approach is a server-side client method. Using newly available elements at the media level, namely the CMAF chunks and with the new “service description” element in the DASH manifest and a hint in the manifest and playlist, both HLS and DASH standards have well-documented elements to anchor the players to an absolute time (UTC timing).

For regular latency services (not based on LL CMAF) players were most likely in best effort mode with regards to stream latency, as described above. However, the low-latency services mandate some elements to give the service provider control over the overall latency for any standard compliant player. As the server-to-client latency is well under control with good precision, the client-to-client relative latency, which is the key element for Watch Together applications, is also naturally well controlled.

Looking at the details, the solution for low-latency services is a bit different between HLS and DASH, but both are based on some common rules:

- Both make use of a timeline to announce the availability of segments.
- Both use CMAF chunks, which can be seen as smaller entities (usually smaller than one second in duration) inside a segment.
- Both have tools to drive exactly the media presentation time, thus not letting the player decide on its buffer level.
- The target overall latency can be defined by the service provider to make sure that whatever the network conditions are all the players will be aligned.

Segments built in compliance with CMAF standards and using CMAF chunk elements can be common, but the manifest and playlist have different elements to signal timing-related information.

The DASH solution is described in the DASH Industry Forum Guidelines (6) and its low-latency extension (7). A new element called "service description" combined with a mandated "producer reference time" information to be set either in the media file (prtf box) or in the manifest gives the unambiguous timing anchors for the segments:

- The "service description" is a new element present in the manifest, giving among other information a target latency (as well as min/max boundaries) for the service. This target latency has to be followed by the player if the value is compatible with what the delivery network can do. It's operator's decision to set a more or less aggressive target based on their knowledge of the targeted device and networks.
- "Producer reference time" information gives the actual time when the content has been either produced or presented at the final ABR encoder input.

With that, a compliant low-latency DASH player can combine the "producer reference time" for a given frame with the target latency to know exactly when this frame should be displayed.

The HLS solution for low latency is based on a different approach to distribute the CMAF chunks, but it also provides a very precise media timeline indication. The part files availabilities are provided directly in the playlist, and the #EXT-X-PROGRAM-DATE-TIME: Tag provides wall clock information of when the first frame of the following segment in the playlist was, for example, entered into the encoder. This information is close to the "producer reference time" mentioned in the DASH specification and can therefore be the exact value of the prtf box contained in the segment.

The PART-HOLD-BACK Tag gives, in a low-latency playback mode, the closest position compared with the live edge (given by the last element in the playlist) where the client can play. This tag can, in combination with EXT-X-PROGRAM-DATE-TIME, determine the target latency the operator has set for the service.

Tests Results

Figure 4 illustrates what can be achieved with a low-latency service delivered with HLS or DASH.

- The desktop player at the top, running on a Chrome browser, is a Dash.js-based player where target latency has been set to 6.5 seconds (DASH stream).
- The player at the bottom left is an Android Exoplayer-based player, running on a Galaxy S6 tablet, playing the same LL DASH stream.
- The player at the bottom right is an HLS player using AVPlayer running on an iPad Pro.

As this can be observed, the three players are in-sync together with an accuracy better than 100 ms. There is no synchronization protocol or interaction between the player except the information contained on the streams.

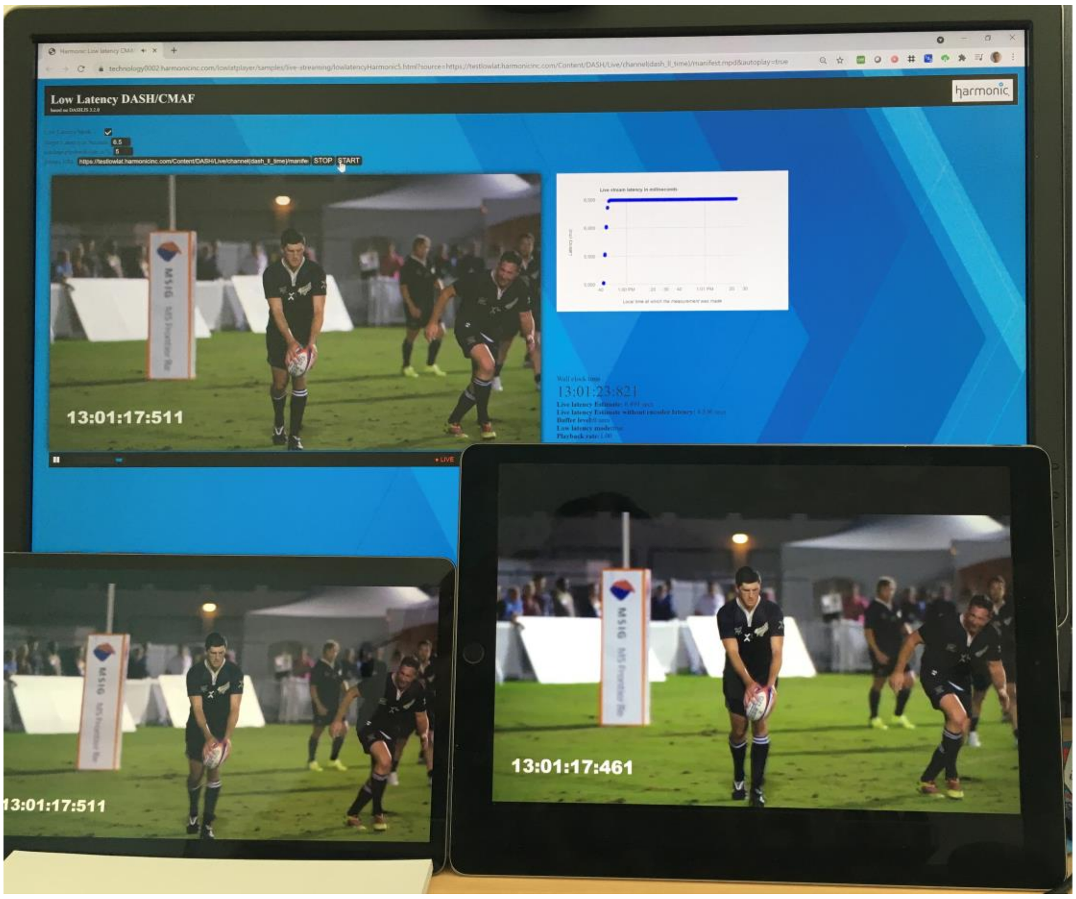

Figure 4 - Latencies on different players

The test above was made with collocated devices but the same kind of tests can be made with geographically distributed devices. Similar experiences with players running HLS or DASH streams, one in Europe and other in the U.S., didn't reveal any noticeable difference in latencies (below a second).

These initial results confirm what low-latency CMAF-based DASH and HLS extensions can provide and show that there is a simple way to provide a Watch Together application for live sport events, for example.

CONCLUSIONS

This paper presented a new approach to guaranteeing proper synchronization between the users of a Watch Together session, using the available tools provided by low-latency HLS and DASH, based on CMAF.

We believe this approach has many advantages for sports event coverage, as these events offer a much better user experience when delivered in low latency. Indeed, this is a fully standardized approach that any player compliant with the low-latency extensions of DASH or HLS can support. By using the time anchor elements, any player, anywhere, will display the same content (same point in time in the media timeline), with an accuracy that can easily be below one second, thus avoiding the need for any client-specified adaption. We believe that creating a Watch Together application is simplified with a focus made on the user interaction capabilities more than lower lever inter-player video synchronization. In addition, no computing power is required on the server side as opposed to a client-based solution.

Moreover, using the low-latency HLS or DASH standard makes the end-to-end latency on par with the broadcast stream, in the range of about five to seven seconds behind live, which is highly valued by OTT service providers streaming sports events.

Watch Together applications are still in their early days and will move progressively from fully proprietary solutions to more feature-rich, standard-based solutions. Lower-level signal synchronization can be achieved thanks to CMAF low-latency delivery and free the development resources for UX improvements and enable a better user experience, which should be the differentiating factor for any of the next-generation immersive applications.

REFERENCES

1. "Co-Watching: Creating the Power of Togetherness," https://www.sportsilab.com/cowatch-ppi
2. ISO/IEC 23000-19:2020, Information Technology — Multimedia application format (MPEG-A) — Part 19: Common media application format (CMAF) for segmented media, https://www.iso.org/standard/79106.html.
3. ISO/IEC 23009-1:2019, Information Technology — Dynamic Adaptive Streaming Over HTTP (DASH) — Part 1: Media Presentation Description and Segment Formats (4th Edition), https://www.iso.org/standard/79329.html
4. HTTP Live Streaming 2nd Edition draft-pantos-hls-rfc8216bis-08. https://datatracker.ietf.org/doc/draft-pantos-hls-rfc8216bis/
5. WebRTC, https://webrtc.org/
6. Guidelines for Implementation: DASH-IF Interoperability Points V4.3: https://dash-industry-forum.github.io/docs/DASH-IF-IOP-v4.3.pdf
7. Low-Latency Modes for DASH : https://dash-industry-forum.github.io/docs/CR-Low-Latency-Live-r8.pdf

ACKNOWLEDGEMENTS

The following individuals are acknowledged for their contributions to the paper: Thierry Fautier at Harmonic for encouraging me to write this paper and providing a review.