Towards Mass Deployment of CMAF

Robert Peck, Jordi Cenzano, Xiangbo Li, Yuriy Reznik
Brightcove, Inc.
Seattle, WA
{rpeck, jcenzano, xli, yreznik}@brightcove.com

Abstract - *Since its inception, the CTA WAVE project and its members have made significant progress in bringing CMAF technology closer to mass deployment. There are now finalized and published Content and Web Media API specifications, as well as operational test suites and clients capable of playing CMAF content. In this paper, we discuss the benefits that adoption of CMAF will bring, as well as challenges that the implementation community may still face in updating their existing OTT media publishing workflows to support CMAF. We will share specific examples and experiences that the engineering team at Brightcove had in building support for CMAF.*

INTRODUCTION

Common media application format (CMAF), also known as MPEG-A Part 19 or ISO/IEC 23000-19 [1], is an MPEG standard, developed in 2015-2017, designed with the goal of unifying media formats used by different HTTP-based streaming systems, such as HLS [2] and MPEG DASH [3]. Perhaps not surprisingly, the timeline for development of this specification has coincided with Apple's decision to adopt ISO base media file format [4] (informally known as "MP4" or "fragmented MP4" or "fMP4") as a container for HLS [6], making it "almost compatible" DASH. Thus, the CMAF development has captured the opportunity to bring both systems together.

The benefits of unifying container formats for HTTP streaming can be easily grasped by looking at typical delivery workflows, presented in Figures 1 and 2 respectively. Figure 1 shows OTT streaming system using 4 different streaming formats: HLS, DASH, Smooth streaming [7], and HDS [8]. In order to publish a given content item, this system runs a cloud transcoder producing 4 different sets of renditions, prepared according to each streaming delivery format. Then all such 4 sets of encodings are used for delivery through a CDN. Naturally, this increases delivery costs, as more content needs to be transmitted. This also diminishes the efficiency of the CDN, as multiple versions of the same encoded content will now need to compete for CDN cache space at the edge servers. Figure 2 shows a system enabling delivery in both DASH and HLS, but using CMAF as a common format. Only one set of encodings needs to be produced and delivered through a CDN. This reduces the costs, increases effectiveness of the CDN, and hence it also increases quality that can be appreciated by end users.

While, as shown in this example, the benefits of CMAF are undisputable in a scenario when it is universally supported by all receiving devices, in practice, the adoption of a new standard may take a long time. In order to promote and foster adoption of CMAF a CTA WAVE project was created [9,10]. The first meeting of CTA WAVE member companies occurred during CES 2016, and over the course of the last 2 years this group has made remarkable progress in moving industry towards transition to this new standard.

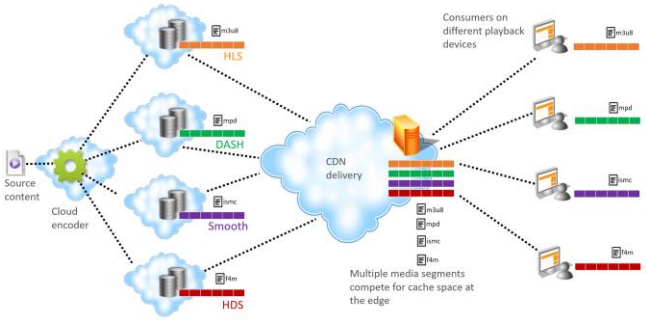

FIGURE. 1: MULTI-PLATFORM OTT WORKFLOW WITH MULTIPLE FORMATS.

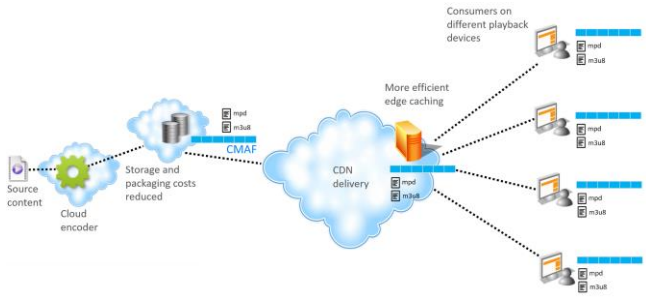

FIGURE. 2: MULTI-PLATFORM OTT WORKFLOW WITH CMAF.

In this paper, we will describe some key elements of the CMAF design, explain how they map to HLS and DASH implementations, additional limits imposed by CMAF and WAVE Content Specification [11], and measures that one needs to take to make sure that content that is being encoded today can be transmuxed to CMAF. We will further discuss an approach that Brightcove took for gradual deployment of CMAF, considering that only a subset of devices can play it currently.

CMAF OVERVIEW

The Common Media Application Format (CMAF) for segmented media [1] is an extensible standard for the encoding and packaging of segmented media objects for delivery and decoding on end user devices in adaptive multimedia presentations. Delivery and presentation are abstracted by a *hypothetical application model* that allows a wide range of implementations including HLS [2] and DASH [3].

The CMAF specification defines several logical media objects:

- *CMAF Track*: contains encoded media samples, including audio, video, and subtitles. Media samples

are stored in a CMAF specified container derived from the ISOBMFF [4]. Media samples may optionally be protected by MPEG Common Encryption [5]. Tracks are made up of a *CMAF Header* and one or more *CMAF Fragments*.

- *CMAF Switching Set*: contains alternative tracks that can be switched and spliced at *CMAF Fragment boundaries* to adaptively stream the same content at different bit rates and resolutions.
- *Aligned CMAF Switching Set*: two or more CMAF Switching Sets encoded from the same source with alternative encodings; for example, different codecs, and time aligned to each other.
- *CMAF Selection Set*: a group of switching sets of the same media type that *may include alternative content* (for example, different languages or camera angles) or alternative encodings (for example, different codecs).
- *CMAF Presentation*: one or more presentation time synchronized selection sets.

Generally, CMAF presentation is the first point where different media types can be combined. In addition to above objects, CTA WAVE content specification [11] also defines

- *WAVE program*: a sequence of CMAF Presentations with consistent encoding constraints enabling continuous rendering.

The *CMAF Hypothetical Reference Model* defines how tracks can be delivered, combined, and synchronized in *CMAF Presentations*, allowing many possible compatible implementations. It is thus possible to create HLS Playlists and a DASH Media Presentation Description, that share the same resources, *CMAF Addressable Objects*, thereby allowing efficient caching even when delivering to multiple platforms.

CMAF Addressable Media objects consist of:

- *CMAF Header:* contains information that includes information for initializing a track.
- *CMAF Segment:* A sequence of one or more consecutive *fragments* from the same track.
- *CMAF Chunk:* A chunk contains a sequential subset of samples from a fragment.
- *CMAF Track File:* A complete track in one ISOBMFF file.

We note that CMAF track file is logical concept, as CMAF assumes that actual deployment workflow may implement late binding, and that fragments do not have to be stored in a single physical file. How it should be stored and in which order things should be processed is left to implementations.

We illustrate the intended model of usage of CMAF objects in Figure 3.

MAPPING BETWEEN CMAF, HLS, AND DASH

As it becomes clear from the above survey, the terminology and meaning of some terms in CMAF do not correspond exactly to terms used by HLS or DASH, or even ISOBMFF. Hence, in order to explain what is actually meant by each CMAF addressable media objects, in Figure 4 we show their intended mapping to boxes in ISOBMFF.

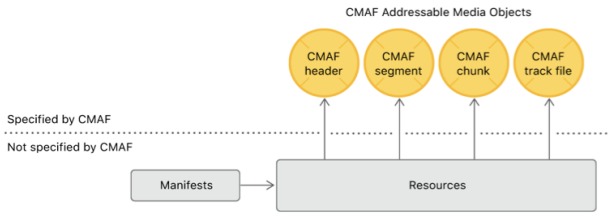

FIGURE. 3: MODEL OF USAGE OF CMAF ADDRESSABLE MEDIA OBJECTS.

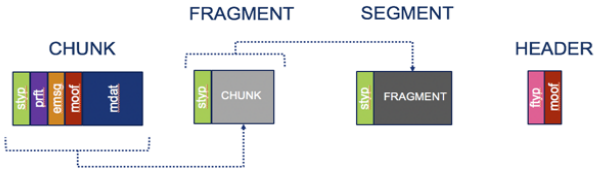

FIGURE. 4: CMAF ADDRESSABLE MEDIA OBJECTS.

CMAF	HLS	DASH
Manifest	HLS Master Playlist and associated Media Playlist (.m3u8) files	Media Presentation Description (.mpd) file
Presentation	Presentation defined by HLS Master Playlist and associated Media Playlists with aligned start points.	DASH Period and associated Adaptation Sets defined in MPD.
Selection set	Sets of parallel tiers of Media Playlists defined by appropriate sets of EXT-X-STREAM-INF tags. Such tiers could be defined, e.g. for different codecs.	A group of Adaptation Sets defined for each Period in MPD.
Switching set	A set of Media Playlists or Variant Streams that can be used by player to play presentation.	DASH Adaptation Set
Track	HLS Variant Stream (specified by Media Playlist URI and EXT-X-STREAM-INF tag describing its properties), restricted to single media type	DASH Representation restricted to single media type.
Header	Media Initialization Section, defined by EXT-X-MAP tag	DASH Initialization Segment
Segment	Sequence of fMP4 segments within same variant stream	Sequence of DASH segments within same representation
Fragment	HLS fMP4 segments limited to single media type (i.e. audio or video)	DASH segment limited to single media type
Chunk	Chunk of fMP4 segment limited to integral number of samples	DASH subsegment
Presentation profile	Only unencrypted or 'cbcs' encrypted profile are supported	Unencrypted, and multiple types of encrypted profiles are supported.

TABLE. 1: MAPPING BETWEEN ELEMENTS OF CMAF, HLS AND DASH.

In Table 1, we further explain a possible mapping between CMAF objects and their implementations in HLS and DASH. We immediately note that this mapping is not symmetric. For instance:

- CMAF presentation could be described as Period in DASH, but DASH, in principle, allows multiple periods, while CMAF, in its scope is restricted to only one.
- CMAF tracks, segments, and fragments can only include single media types (e.g. audio or video), while in HLS or DASH one can multiplex multiple media types in the same segments.
- CMAF defines the concept of Chunks, which can be mapped to subsegment in DASH terminology, but not something that is specifically defined in HLS.
- CMAF defines several types of encrypted presentation profiles, of which HLS only supports 'cbcs' profile.

What practically all these differences mean is that encodings of streams with the intent of packaging to CMAF have to be done somewhat differently. Separate files have to be created for each data type. Encryption should be with 'cbcs' or delayed till the point when it is understood which manifest is used. In cases when low delay is desired, chunk-based encoding should be used, producing addressable fragments and chunks as specified by CMAF.

There are, however, also many additional encoding constraints can also be inferred from text of CMAF [1], as well as WAVE content specification [11].

CMAF AND WAVE PROFILES

WAVE content specification [11] defines several media and presentation profiles, summarized in Tables 2, 3, 4, and 5.

Table 2 presents WAVE video profiles. The WAVE "HD" profile coincides with HD profile defined in CMAF specification [1], Table A1, which further defines maximum resolution of 1080p and maximum framerate of 60fps.

Table 3 presents WAVE audio profiles. The WAVE "AAC Core" and "AAC Adaptive" profiles coincide with same-named profiles defined in CMAF specification [1], Table A2, where "AAC Adaptive" is understood as a constrained subset of "AAC Core", enabling adaptive switching (this relates to the fact that HE-AAC and HE-AAC v2 profiles enable additional tools requiring longer delays than AAC-LC, and unless the encoder does extra work to mask the their effects in the vicinity of segment boundaries, the switching may produce audible artifacts).

Table 4 presents WAVE profiles for subtitles and captions. It is much narrower than subtitle profiles allowed by Table A3 in the CMAF specification [1], and certainly narrower than current subtitle support in both HLS and DASH. Basically, what is intended to be used are TTML IMSC1 subtitles, and gone are CAE 608/708 and WebVTT.

Finally, Table 5 presents 3 presentation profiles currently defined by the WAVE content specification [11]. All 3 profiles have the same constraints to audio, video, and subtitles, but differ in Encryption scheme. The HLS-based implementations, should be able to use CMFHD and CMFHDs profiles, while DASH-based implementations may also be able to use CMFHDc.

Media Profile	Codec	Codec Profile	Level	Color	Transfer	Brand
HD	AVC	High	4.0	1 (709)	1 (709)	'cfhd'
HHD10	HEVC	Main10 MainTier	4.1	1 (709)	1 (709)	'chh1'
UHD10	HEVC	Main10 MainTier 10-bit	5.1	1 (709) 9 (2020)	1 (709) 14 (2020)	'cud1'
HLG10	HEVC	Main10 MainTier 10-bit	5.1	9 (2020)	18 (HLG) 14 (2020)	'clg1'
HDR10	HEVC	Main10 MainTier 10-bit	5.1	9 (2020)	16 (PQ)	'chd1'

Notes: 709 = ITU-R BT.709, 2020 = ITU-R BT.2020, HLGPQ= ITU-R BT.2100

TABLE. 2: WAVE VIDEO PROFILES.

Media Profile	Codec Family	Codecs Profiles	Level	Brand
AAC Core	AAC	AAC-LC, HE-AAC or HE-AAC v2	2	'caac'
Adaptive AAC Core	AAC	AAC-LC, HE-AAC or HE-AAC v2	2	'caaa'
AAC Multichannel	AAC	AAC-LC, HE-AAC	6	'camc'
Enhanced AC-3, including AC-3	AC-3 EAC-3	AC-3 EAC-3	n.a.	'ceac'
AC-4, Single Stream	AC-4	AC-4	3	'ca4s'
MPEG-H, Single Stream	MPEG-H	LC	3	'cmhs'
DTS-HD Audio	DTS-HD	DTS, DTS-HD	n.a.	'dts1'
xHE-AAC	USAC	xHE-AAC	4	'cxha'

TABLE. 3: WAVE AUDIO PROFILES.

Media Profile	Description	Brand
TTML IMSC1 Text	IMSC1 Text Profile	'im1t'
TTML IMSC1 Image	IMSC1 Image Profile	'im1i'
TTML IMSC1.1 Text	IMSC1.1 Text Profile	'im2t'
TTML IMSC1.1 Image	IMSC1.1 Image Profile	'im2i'

TABLE. 4: WAVE SUBTITLES PROFILES.

Presentation Profile	Required Video Media Profile	Required Audio Media Profile	Required Subtitle Media Profile	Allowed Encryption Scheme
CMFHD	'cfhd' HD AVC	'caac' AAC Core	'im1t' TTML IMSC1 Text	None
CMFHDc	'cfhd' HD AVC	'caac' AAC Core	'im1t' TTML IMSC1 Text	'cenc'
CMFHDs	'cfhd' HD AVC	'caac' AAC Core	'im1t' TTML IMSC1 Text	'cbcs'

TABLE. 5: CMAF PRESENTATION PROFILES.

CMAF AND WAVE ENCODING CONSTRAINTS

CMAF specification [1] further defines a number of constraints that compressed audio and video streams must satisfy. For example, for video codecs, this includes:

- Track-level constraints:
  - Same bit-depth across tracks
  - Same chroma sampling formats
  - Must include VUI parameters
  - Must have continuous frame numbers
- Switching-set-level constraints:
  - All tracks must have same DAR, although SARs and frame sizes may differ
  - All tracks must be encoded using same source, same color space, transfer function, color primaries, color volume, brightness, bitdepth, and presentation timing
  - All tracks in a switching set must have same initialization constraints (CMAF headers).

The Annex A of WAVE content specification [11] provides some additional constraints that are intended to be used across WAVE programs:

- All presentations must have *consistent picture aspect ratio* (which effectively means same SAR and DAR for all tracks)
- All presentations should have *consistent framerate*
- Must have same audio channel layout
- Must have consistent video color characteristics.

In summary, in many aspects, CMAF and WAVE specifications appear to be more restrictive than existing deployment guidelines for HLS or DASH. But the next question that arises is – how one can enable delivery to CMAF decoding capable devices while also supporting HLS and DASH delivery to legacy ones?

To provide one possible answer, in the next section we will describe architecture of a multi-platform OTT system that was developed by Brightcove.

BRIGHTCOVE VIDEOCLOUD PLATFORM

In Figure 5 we present high-level architecture of Brightcove VideoCloud platform [12]. This is a cloud-based media delivery system, including multiple functional components, and assuming certain order of events in its publishing chain.

I. Job request, ingest and transcoding

The request to publish content is usually made by an operator, who also places the content on a certain origin server and provides its URL as part of the request.

The ingest and transcoding of the content is subsequently done by a cloud transcoder (in this case Zencoder [13]). There are actually 2 steps in the transcoding operation. First is an *encoding profile generation*, which produces a ladder of resolutions, rates, and other codec constraints to be applied to all renditions forming ABR adaptation set (or CMAF switching set) for a given content. This step is done by a component which we call Context Aware Encoding or CAE [14-16]. Once encoding profiles are created, all renditions are subsequently produced by the cloud transcoder. Profile generation and transcoding is done separately for audio and video tracks, and pluralities of such streams along with additional metadata are then stored in internal storage that belongs to the dynamic delivery system.

II. Device detection and manifest generation

The *dynamic delivery system* is essentially a layer orchestrating selective transmuxing, encryption and placement of content on CDNs. It is effectively driven by player's requests. Once the player sends a request for media though a playback API, the dynamic delivery system generates a list of manifest URLs, representing all possible permutations of delivery protocols (HLS, DASH, Smooth, etc.), file formats, codecs, and DRMs that may be supported by the receiving devices. HLS and DASH manifests relying on CMAF as a common format represent a subset of this list.

After the player receives the list of URLs, it picks the one that fits the best business and technical requirement and then pulls corresponding manifest from CDN. If the manifest is not in the CDN yet, the dynamic delivery system will be invoked and the missing manifest will be generated, and delivered through CDN to the player.

In cases when a requested manifest has not yet been generated, the dynamic delivery system first tries to identify the type and capabilities of a device that is requesting it. This process is called *device detection*. The list of properties that device detection is trying to establish is shown in Table 1.

Property	**Possible values**
Device type	PC, smartphone, tablet, TV, etc.
OS type / version	Android 6.0, iOS 11, etc
Browser type/version	Chrome 51, Mozilla 5.0, etc
Geographic region of device	Country code
Video codec support	H.264 baseline, H.264, HEVC, etc.
Supports codec switching	Yes/No
Maximum supported resolution	1080p, 540p, 480p, etc.
Maximum supported bitrate	128kbps, 1.2Mbps, 10Mbps, etc.

TABLE. 1: EXAMPLE PROPERTIES OF DEVICE.

Once device properties are established, and metadata related to encoded content are retrieved, a manifest targeting this specific device is generated. This is a selective process, where only renditions that match decoding capabilities of the devices are included in the final manifest. Furthermore, the targeting of manifest is controlled by certain rules that the operator may specify. E.g. operator may specify to use different CDNs in different geo locations, or impose different limits on maximum resolutions and rates, etc. Such customization is done by *rules engine*, using *rules API* as the interface to the operator.

III. Just-in-time packaging

When manifest is finally received by the player, it starts retrieving media streams according to URLs specified there. Such media may or may not be present in the CDN. In cases it is missing, the CDN miss response brings control back to dynamic delivery system, which activates just-in-time packager to generate it. To generate such content, the just-in-time packager retrieves segments of previously encoded

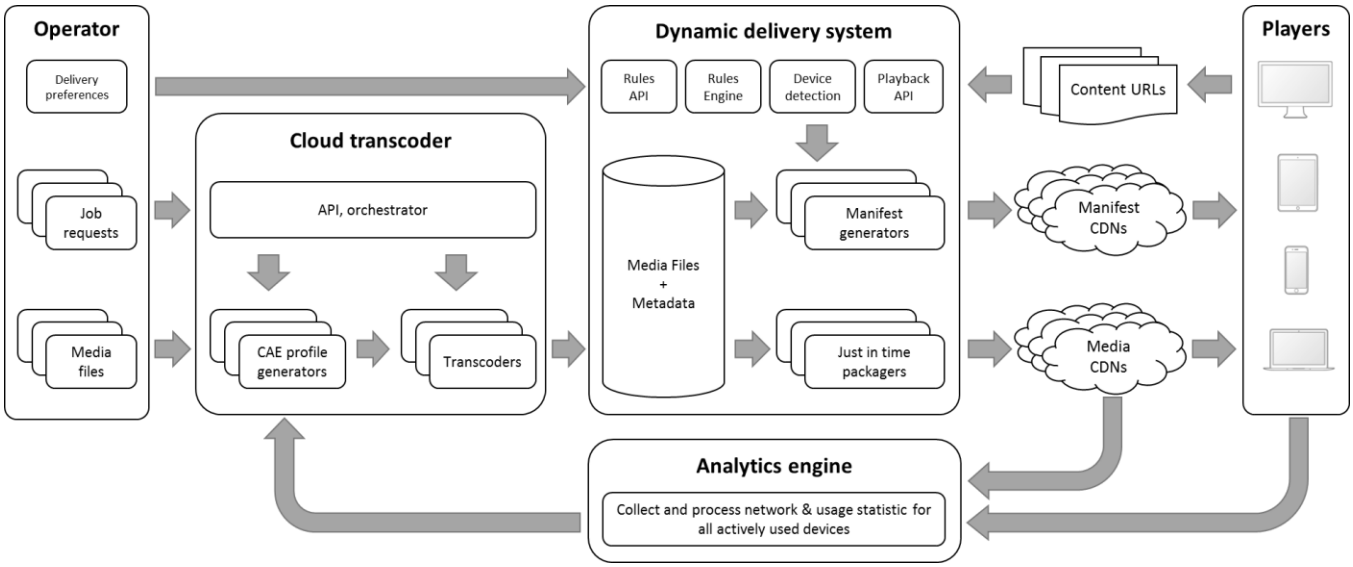

FIGURE 5: HIGH-LEVEL ARCHITECTURE OF BRIGHTCOVE VIDEOCLOUD PLATFORM.

component (audio and video) streams from storage and transmuxes them by converting them to a particular form of ISOBMFF or TS segments as needed for delivery. The intermediate format used for storage and additional metadata make such conversion a fairly light-weight process. Support of CMAF comes here as one of the formats supported by the transmuxer.

Once the missing segment is generated, it is passed back to the CDN, and then subsequently to the player. Note that such transmuxing operation needs to be applied only for content that has not yet been cached by CDN or which is used so infrequently that it got purged.

IV. End-to-end optimizations

The Brightcove VideoCloud system also includes means for end-to-end performance optimizations, such as collection of *bandwidth and usage statistics*, and their use in CAE profile generation. The details about such optimization process and related science can be found in [15-17].

ROAD TOWARDS MASS DEPLOYMENT OF CMAF

As of time of this writing, the list of devices that can support CMAF is fairly limited. According to Apple technical note [18], the support for CMAF on Apple devices starts with iOS 10.0, macOS 10.12, and tvOS 10.0 or later OS versions. To reach majority of desktops, certain modifications are needed in the web players, and such work is currently under way [18].

But even with limited CMAF support as of now, it is clear that some such devices already exist and that they are fully functional. Everything that is needed to support proper CMAF content generation is also already in place. The resources outlined in [18] include detailed content specification, content validator, system conformance suite, etc.

In our experience we have found that adding CMAF to a system that already supports dynamic transmuxing to several existing delivery formats is relatively simple, and boils down to a few elements: more restrictive profile generation and encoding, adding an extra flavor of ISOBMFF transmuxer, and adding extra rules to HLS and DASH manifest generators to produce CMAF-compatible manifests.

While in the short term CMAF most likely will have to co-exist with other varieties of HLS, DASH, and some other delivery formats, the more devices will become capable of decoding it, the clearer benefits we will start to see. Even with dynamic transmux and delivery, the use of CDNs still remains suboptimal, with multiple versions of same content competing for CDN cache at the edge. The more players will start picking CMAF packaged versions, the higher will be probability that such content will be in cache. That will increase quality of experience, and will also lower delivery costs. This is a win-win situation for both publishers and consumers.

REFERENCES

[1] ISO/IEC 23000-19:2018, “Information technology - Coding of audio-visual objects - Part 19: Common media application format (CMAF) for segmented media”. https://www.iso.org/standard/71975.html

[2] R. Pantos and W. May, “HTTP live streaming, RFC 8216,” https://tools.ietf.org/html/rfc8216, August 2017.

[3] ISO/IEC 23009-1:2014, “Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats”, https://www.iso.org/standard/65274.html

[4] ISO/IEC 14496-12:2015, “Information technology -- Coding of audio-visual objects -- Part 12: ISO base media file format”, https://www.iso.org/standard/68960.html

[5] ISO/IEC 23001-7:2016, “Information technology -- MPEG systems technologies -- Part 7: Common encryption in ISO base media file format files”, https://www.iso.org/standard/68042.html

[6] T. Siglin, “HLS Now supports fragmented MP4, making it compatible with DASH”, http://www.streamingmedia.com/Articles/News/Online-Video-

News/HLS-Now-Supports-Fragmented-MP4-Making-it-Compatible-With-DASH-111796.aspx

[7] A. Zambelli, Smooth streaming technical overview, Microsoft Corp., Redmond,WA, USA, Tech. Rep. [Online]. Available: http://www.iis.net/learn/media/on-demand-smooth-streaming/smooth-streamingtechnical-overview

[8] Adobe HDS streaming specifications, https://www.adobe.com/devnet/hds.html

[9] CTA WAVE project, https://cta.tech/Research-Standards/Standards-Documents/WAVE-Project/WAVE-Project.aspx

[10] "Internet Video Leaders Announce Interoperability Effort", Consumer Technology Association, Dec 2, 2015, https://www.cta.tech/News/Press-Releases/2015/December/Internet-Video-Leaders-Announce-Interoperability-E.aspx

[11] CTA-5001, "Web Application Video Ecosystem – Content Specification", April 2018, https://cta.tech/cta/media/EventImages/TechStandards/CTA-5001-Final_v2_pdf.pdf

[12] Brightcove VideoCloud. https://www.brightcove.com/en/online-video-platform

[13] Zencoder https://zencoder.com/en/

[14] Brightcove CAE https://www.brightcove.com/en/context-aware-encoding

[15] Y. A. Reznik, K. O. Lillevold, A. Jagannath, J. Greer, and J. Corley, "Optimal design of encoding profiles for ABR streaming," in Proceedings of the 23rd Packet Video Workshop, New York, NY, USA, 2018, PV '18, pp. 43–47, ACM.

[16] Y. A. Reznik, X. Li, K. O. Lillevold, A. Jagannath, and J. Greer, "Optimal design of multi-codec profiles for ABR streaming," ICME 2019 – submitted.

[17] Y. A. Reznik, X. Li, K. O. Lillevold, R. Peck, and R. Marinov, "Optimizing Mass-Scale Multi-Screen Video Delivery", NAB 2019 – submitted.

[18] "About the Common Media Application Format with HTTP Live Streaming", https://developer.apple.com/documentation/avfoundation/media_assets_playback_and_editing/about_the_common_media_application_format_with_http_live_streaming

[19] CTA WAVE Boot Camp 2018 presentation. https://cta.tech/cta/media/Standards/2018-CTA-Fall-Forum-WAVE-Boot-Camp-final.pdf