Robust SCTE 35 in the OTT workflow

Rufael Mekuria
R&D Group
Unified Streaming
Amsterdam NL
rufael@unified-streaming.com

Yasser Syed
GTO SAT
Comcast
Philadelphia, PA USA
yasser_syed@comcast.com

Gary Hughes
DPI & HAS Technologies
Boston, Massachusetts USA
ghughes1138@gmail.com

ABSTRACT

Dynamic ad insertion and substitution are increasingly popular to monetize OTT services. A key enabling technique is the signaling of timeslots for insertion or substitution. In practice, SCTE 35 has been used for this, but OTT implementations have been somewhat inconsistent in practice. This paper highlights advances in ad insertion from both practical and a standardization perspectives to realize consistent and robust ad slot signalling in end-to-end OTT workflows. Guidelines for server or client-based ad substitution using SCTE 35 in DASH and ISO base media file formatted files are presented. It addresses the SCTE 35 fields, ISO BMFF and DASH fields and their relationship. The guidelines have been recently developed through joint discussions by DVB and SCTE resulting in publications as DVB-TA part 3 and SCTE 214 (2022) (and in its upcoming guidance annex) respectively. Additionally, this paper provides examples and illustrates specific use cases, such as the early termination of ad breaks, insertion of ads, and MPEG DASH period splitting. To showcase usage in a DASH manifest manipulator a cloud function implementation for period splitting of large dynamic presentations in real-time is described. Additionally, this paper details emerging techniques to carry SCTE 35 upstream in ISO BMFF and CMAF timed metadata tracks based on the recently published event message track ISO/IEC 23001-18 specification. This metadata track format mainly targets storage and upstream use cases. An implementation is described that generates example files based on this format with ad slot signaling and it is used to implement a distributed live CMAF uplink.

CCS CONCEPTS

• Multimedia Information Systems

KEYWORDS

Standardization, Streaming, Signaling.

ACM Reference format:

Rufael Mekuria, Yasser Syed and Gary Hughes. 2023. Robust SCTE 35 in the OTT workflow. In *Proceedings of ACM Mile High Video conference (MHV'23). ACM, New York, NY, USA, 6 pages.* https://doi.org/10.1145/3588444.3591000

MHV '23, May 7–10, 2023, Denver, CO, USA

ACM 979-8-4007-0160-3/23/05…$15.00
https://doi.org/10.1145/3588444.3591000

1 Introduction

Over-the-top (OTT) delivery using HTTP Adaptive Streaming (HAS) has become extremely popular for the delivery of media. To complement the increase of *subscription-based* streaming services, new models are emerging supporting *ad-supported* streaming for these services. This enables, in some cases, to realize *free ad supported TV services* (FAST) or reduced-price *subscription-based* services that include advertisements. OTT delivery is rather flexible as it enables both server-side and client-side ad insertion. For linear broadcast services, ad substitution is used to replace existing ads whereas in non-live use cases ad insertion could also be used extending the media timeline with an inserted advertisement. Since the HAS client chooses what segments to download instead of the server, dynamic ad insertion and substitution can be achieved using manifest manipulation (server-side) *or* by specific OTT player behavior (client side). Both approaches to ad insertion and substitution are popular.

Regardless of the approach, a key technique towards supporting dynamic ad substitution and insertion is the signaling of dynamic metadata in live streams and programs. Traditionally this is achieved using SCTE 35 [1] markers based on the corresponding specification, developed by the Society for Cable and Telecommunications Engineers (SCTE), which also enables the splicing of media presentations. SCTE 35 signaling was designed in the 1990s to work with MPEG 2 transport streams (TS) , but (at least initially) not for streaming or ISO Base Media file formatted content. Since then, it has been extensively updated nearly on a yearly basis to support embedding of splicing commands that enable targeted substitution of content in live programs (upstream splicing). For more information, see guidelines developed in SCTE-67 [7]. For adaptive streaming constraints, see SCTE 214 [2] and HTTP Live Streaming (HLS) [4] to provide information and guidance on how to use SCTE 35 in MPEG DASH [3] and HLS [4] respectively. Despite this information, a lot of variations continue to exist resulting in non-interoperable implementations across different streaming platforms. To enable correct splicing functionality in MPEG DASH [3] and other OTT protocols, their specific timing and playback model needs to be taken into account.

HAS is a different distribution mechanism, i.e, it is largely pull based instead of push based, leading to different causality of events and media download/playback. Furthermore, as OTT is converging towards the use ISO BMFF segments instead of MPEG-2 TS segments, fields relating to MPEG-2 TS need a different contextual interpretation. A well-defined interpretation and mapping make the usage of SCTE 35 signaling in OTT streaming explicit. This is key for enabling common use cases of ad slot signaling and splicing and preserving SCTE 35 in OTT workflows.

This paper aims to highlight some of the key interoperability issues encountered when inserting SCTE 35 in OTT workflows. It presents recent developments and discussions in DVB and SCTE to harmonize usage of SCTE 35 to signal ad slots in MPEG DASH. The specifications developed by SCTE and DVB are expected to be helpful for practitioners in the industry to achieve a robust implementation across the OTT workflow.

A second topic this paper will discuss relates to inserting separately SCTE 35 upstream in the OTT workflow such as when using multiple distributed encoders and or streaming servers and storing of SCTE 35 metadata. To this goal, we present advances in MPEG file format for carriage of event messages including SCTE 35 in ISO/IEC 23001-18. The event message track format is very similar to the carriage of web-vtt in MPEG-4 tracks, i.e. using boxes in samples and sample relative timing. We provide some example files and an example implementation to facilitate the implementation of this timed metadata event message tracks carrying SCTE 35 based on ISO/IEC 23001-18. This approach can also be used for both storage and redundant encoding and packaging implementations. By illustrating both techniques for upstream and downstream signaling, we aim to provide a holistic overview of the SCTE 35 signaling options in OTT workflows.

2 Architecture

Figure 1 illustrates a common architecture that embeds server-side and client-side Dynamic Ad Insertion in an OTT workflow [9] based on consensus achieved in the DASH Industry Forum. The Adaptive Bit Rate (ABR) encoder is the entry point to the OTT workflow. In some cases, MPEG-2 TS is used as an input and in other cases contribution/baseband sources such as SMPTE 2110 or HD-SDI signals are used. In some input signal types ad slot related metadata may already exist, such as in the form of SCTE 104 or SCTE 35 messages. In the diagram, the ABR encoder block is in charge of transcoding the media to ABR representations. The output of it is then passed to a streaming server, packager or cloud storage. The streaming server or ABR encoder may connect to a DRM service to request key information.

The next step after the streaming server includes delivery of the media segments and the Media Presentation Description including the SCTE 35 signaling by a Content Delivery Network (CDN). Server-side ad insertion is often implemented using a manifest manipulator, which can be run at the CDN edge or independently. The server-side dynamic ad insertion operation performs calls to an ad server based on the advertised SCTE 35 ad slots and stitches the content into the manifest (e.g. by replacing a period indicated as ad slot without altering the linear content timeline). Server-side ad insertion is transparent to the player which just sees the resulting manifest generated by the manifest manipulator service. In client-side ad insertion, the app/player would be responsible to do the request to the ad server and render the ad in the presentation.

Based on this architecture, it is clear that a robust way to traverse SCTE 35 based ad-slot signaling is required. In particular there are 2 key interoperability points to consider: 1) the upstream signaling from the ABR encoder and 2) the downstream signaling to the manifest manipulator or player. The upstream signaling can be implemented through a SCTE 224/250 ad decision signaling approach or through a format based on timed metadata tracks which will be described later in this paper. The downstream signaling can applied in the manifest and is detailed further in this paper. Inconsistent signaling can potentially interrupt ad insertion solutions, causing no ad to be replaced, or even break the playback.

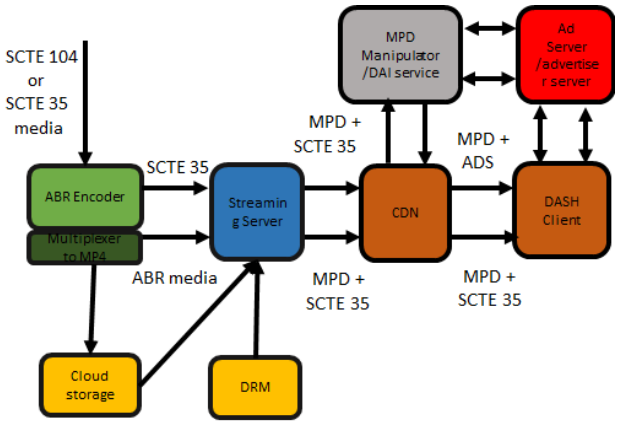

Figure 1 OTT workflow for server-side DAI

3 Related Work and Literature Review

Apart from the techniques outlined in Figure 1 in the previous section, different approaches exist for loading advertisements onto a player. For example, loading an IAB VAST [11] document directly at the player is one option. IAB VAST defines an XML format that can contain different media sources and different tracking and advertisement elements that enable a VAST client/player to play combined media presentations with both main and ad content. For live and broadcast streams, the player may do a VAST request during playout and before an ad break becomes active. In this case the VAST response would contain an Inline element with ad content that may be stitched into the linear content.

Another approach already existing is based on SCTE 130 [6] which is a set of standards for ad decisioning systems ,developed by SCTE, to achieve similar functionalities. In practice VAST [11] and SCTE 130 [6] may be used for either server-side ad insertion or client-side ad insertion and in live streams SCTE 35 slots may trigger servers or clients to do the actual requests to the ad servers. In server-side ad-insertion the media player or client may be totally unaware of VAST elements such as tracking elements. For real-time decisioning for SCTE methods, SCTE 250 [5] defines an Event Signaling and Management (ESAM) interface, with this API an entity (encoder or packager) can do a request to a signal decisioning service (SDS), that can decide if an ad slot or metadata is valid and should be used for the output or not.

SCTE 35 [1] is a format for carrying programme metadata and splice information, it defines the *splice_info_section()* that can carry *splice_commands* and *splice descriptors*. The most common splice commands for ad slot signaling are *splice_insert* and *time_signal* combined with *segmentation_descriptor*. A *Splice_insert* command signals a splice in or out of the network (i.e. the main program) or both when an *auto_return*=1 which is used for returning back to the main program at the end of the break automatically. The *time_signal* command is a bit more complex, in this case it is expected that one or more segmentation descriptors are appended to identify the exact semantics of the segmentation. In practice, *time_signal* requires explicit start and end segmentation descriptors to mark the out and in splices carried in different

markers and thus enable hierarchical segmentation. This makes usage of *time_signal* tricky, especially in HAS cases that benefit from knowing the exact duration of a segment in advance. HTTP live streaming defined in RFC 8216 [4] support encapsulation of *splice_insert* or a *time_signal* pair by using #EXT-X-DATERANGE spanning from time of the out to the time of the in and carrying the entire splice info section encoded in Hexadecimal, including the *splice_insert* or *time_signal* pair. In SCTE 214 [2], the SCTE profile for DASH defines the encapsulation of SCTE 35 message in an MPEG DASH Event.

The usage of SCTE 35 for the hybrid scenario where the main content is delivered over broadcast and the ad content video is fetched from an Internet connection is specified in DVB-TA [12] part 1 with specific constraints on the implementation e.g. upid type to enable consistent signaling of identifiers related to the broadcast. The specification(s) adds more constraints to enable consistent implementation of the receivers, that can do a client- based ad insertion/replacement. The Addressable TV specification [10] implements such scenarios and is deployed in France. Similarly, a consortium of Dutch operators developed a similar national specification for embedding ad slots using SCTE 35 [14]. Some solution using the HbbTV platform [19] have been proposed with similar features. All these scenarios combine the broadcast signal for the main content and Internet for the ad content using client-side ad insertion.

For OTT, server-side manifest manipulation is currently more common as presented in [18], but ideally both cases are to be supported for ad slot signaling as more TV and streaming services move to OTT delivery for the main content and the ad content. DVB-TA part 3. In SCTE 214 and its upcoming annex, improved support for such use cases for DASH main content is presented to enable both server and client/player side ad insertion.

4 SCTE 35 in DASH

4.1 SCTE 35 constraints

Carriage of SCTE 35 markers in DASH is defined in SCTE 214 [2] and uses MPEG DASH `EventStream` and `Event` elements. In this case when SCTE 35 information is used for signaling of splicing information, it becomes increasingly important to constrain some fields, both in the SCTE 35 payload and in DASH. Further constraints at both levels helps to standardize some of the interpretation of these fields by a receiver. DVB-TA part 3 [13] was developed to document constraints, complementing DVB-TA part 1 [12] to support DASH for client *and* server based ad insertion. Some of the more general constraints have also been included in a recent update to SCTE 214 for overall improved compatibility. In this section we detail the recommendations from DVB-TA part 3 [14], which is the profile intended to be used with DVB-DASH.

For ad slot signaling in DVB-TA, currently only command type 5 *splice_insert* and 6 *time_signal* are supported. For all DASH based signaling the time of the `Event`@presentationTime is necessary for determining the splice time on the DASH presentation timeline. This implies that fields like *ptsAdjust, pts_time* etc. may be ignored by receivers. In some cases, they may still be non-zero in case there is a need to convert back to an MPEG-2 TS downstream or for other reasons. For *splice_insert* command (type 5), DVB-TA part 3 [14] requires *splice_event_cancel_indicator=0* as cancellation is not supported. *Splice_event_id* should be (pseudo) unique and cannot be used to correlate in and out markers as this correlation can simply be applied by a time based correlation (looking for the prior *splice_insert*). The *program_splice_flag* and *duration_flag* must both be set to 1(true). The *splice_immediate* flag is not constrained, but given limited implementation and support of *splice_immediate* in legacy implementations, it recommended to set it to 0 to improve compatibility. For *splice_insert* when *out_of_network_indicator*=1 (out marker) the duration shall be set to the expected break duration. The duration shall be 0 when *out_of_network_indicator*=0 (in marker). For the *time_signal* command similar recommendations apply, i.e. *splice_time*() may be set but would typically be ignored by a player. For the segmentation descriptors, that are usually added to the *splice_info_section and are* carrying the *time_signal* command, similar constraints are defined as for the *splice_insert case*. It is expected that usage of *time_signal* is combined with *segmentation_descriptors*. Currently, DVB-TA part three only supports *segmentation_type_id* 0x34, 0x35, 0x36, and 0x37 for ad insertion/substitution. *segmentation_duration* equals the expected break duration for a start descriptor, or 0 for end descriptor (both in the marker as in the MPEG DASH `Event` Element). The default signaling targets ad substitution (i.e replacement), but in OTT cases ad insertion is sometimes also required. DVB-TA part 3 specifies insertion ad breaks with a *splice_insert* with out of network indicator set to 1 and a duration of zero or a *time_signal* with a segmentation descriptor start with *segmentation_duration* of zero and an end descriptor in the same payload with a corresponding segmentation_event_id. This type of signaling is specific to OTT or video on demand cases when an advertisement can be inserted to extend the media timeline. This functionality was not traditionally covered in SCTE 35, but some of the server-side DAI OTT ad services already implemented this functionality. DVB-TA part 3 also supports early termination using an earlier than expected *splice_insert* or end marker, but this requires some additional constraints on the service as outlined in the next section. Many of the SCTE 35 fields that do not relate to the timing or other operational aspects are not further constrained in DVB-TA part 3, as these fields (e.g. upid type) may be specific to the ad insertion solution or a higher level application standard.

4.2 DASH EventStream constraints

To signal the markers in DASH, it is recommended to use an `EventStream` in the media presentation description with @schemeIdURI=*"urn:scte:scte35:2014:xml+bin"* as specified in SCTE 214 [2]. In this scheme the SCTE 35 *splice_info_section* is base64 encoded and inserted in a snippet of XML in the MPEG DASH media presentation description `Event` element. When generating DASH events, the event's presentation time corresponds to the SCTE 35 splice time, and timing information must be converted to the DASH timeline. For video tracks the `Event`@presentationTime shall correspond to the start of a media segment. For audio tracks, slight inaccuracies up to 100 ms are tolerated to account for slight misalignment between audio and video segments. To achieve this accuracy between timelines, careful selection of segment duration and EventStream@timescale are critical. `Event`@duration is set to the duration corresponding to the SCTE-35 maker as described in previous section. To generate unique `Event`@id it is recommended to use a checksum of the payload or another unique identifier. In addition some other issues have been reviewed and addressed.

One issue, that often comes up when using SCTE 35 in DASH is how to enable global timing in MPEG DASH, such that a SCTE 35 marker relating to a certain time of the program can be signaled, even in a new period. `Event`@presentationTime is used to define the splice point, even if it is in a new Period but in MPEG DASH the timing of `Event`@presentationTime is relative to the beginning of the period. In MPEG DASH 4th edition [3], the `EventStream`@presentationTimeOffset attribute was introduced. This attribute sets the presentationTime of an event that matches the start of the Period. For example for an `EventStream`@presentationTimeOffset="1000" and an `Event`@presentationTime="1020", the event starts 20 ticks on the timescale after the `Period`@start. By setting this attribute, for example to a value corresponding to the `Period`@start time, event timing can be made continuous across periods. Another aspect for enabling robust SCTE 35 signalling in DASH for dynamic ad insertion, considers presence in the media presentation descriptions. DASH MPD with @type="dynamic" use MPD updates and `EventStream` elements where Event elements may be removed or added. If this is done inconsistently, some implementations that rely on this information being present may break. Therefore, the guideline recommends Events to remain present in the media presentation description (or in the media segment) as long as they are active and even an extended period after that, i.e. as long as overlapping segments are in the media presentation description and available. Improving the signaling of MPD events was done in coordination with DASH-IF leading to the DASH-IF IOP version 5 part 10 [9] that details consistent signaling of Events in dynamic media presentation descriptions. This consistency in event signaling facilitates correct time shifts and seeking behavior.

Table 1 Example SCTE 35 in Event and parsing

<Period id="1519" start="PT451209H39M31.000S">

<EventStream schemeIdUri="urn:scte:scte35:2014:xml+bin" timescale="1" presentationTimeOffset="1624354771">

<Event presentationTime="1624354848" duration="19" id="760">

<Signal xmlns="http://www.scte.org/schemas/35/2016">

<Binary>/DAgAAAAAAAAP/wDwUAAAL4f//+ABoXsMAAAAAAAPF20V0=</Binary>

</Signal>

</Event></EventStream>

<!!—AdaptationSet omitted →</Period>

```
"splice_command_type": 5,
 "splice_command":
   "splice_event_id": 760,
   "splice_event_cancel_indicator": false,
   "out_of_network_indicator": true,
   "program_splice_flag": true,
   "duration_flag": true,
   "splice_immediate_flag": true,
   "splice_time": null,
   "break_duration":
    "auto_return": true,
    "duration":
      "pts_time": 1710000,
      "wall_clock_seconds_ext": 19.0,
      "wall_clock_time_ext": "00:00:19:00000"
```

DVB-TA further developed the common use case of early termination from an ad break. For this, guidelines on how the segments are published in the MPD (i.e. segment by segment) to keep alignment between the main content and ad are provided, such that the early termination/switch back can be implemented seamlessly. A key attribute to enable this is the @maxSegmentDuration that gives an indication of the maximum segment duration used defining a granularity for making an early return. By not announcing many segments in the manifest ahead, early termination can be achieved without breaking player behavior and newly defined player behavior is not required.

Events that signal an early termination of an ad break shall remain in the manifest at least as long as the original ad break signal is in the media presentation description/manifest to avoid inconsistency, and the Event@duration and SCTE-35 duration is zero.

In *Table 1,* we provide an example of respective signaling of a SCTE 35 marker as an MPEG DASH `Event` in an `EventStream` element. In this case, EventStream@presentationTimeOffset is used, so the start time of the splice point is 1624354771 – 1624354848 = 77 seconds after the start of the Period in this case. The marker itself is base64 encoded, in the table we show some of the decoded fields showing this is a *splice_insert* with *auto_return* for a 19 second break and correctly set flags. The AdaptationSet with `SegmentTemplate` and `SegmentTimeline` is not shown for brevity in the example. The `EventStream`@timescale of 1 may work in this example, but in most cases a larger timescale is required for frame accuracy. Another case that was discussed was the case of multiple segmentation descriptors and selecting the corresponding `Event`@duration, it is recommended to always chose the duration of the *longest* segmentation descriptor for these cases.

4.3 Use case: period splitting

SCTE 35 Events, when formatted according to DVB-TA recommendations can be used to split a single period MPD into multiple periods. The algorithm consists of the following method along with the recommended input as a DASH MPD that uses the iso live profile with `SegmentTemplate` and `SegmentTimeline` using *splice_insert* with *auto_return*=1.

1. Detect SCTE 35 commands splice_insert and *out_of_network* indicator = true and *auto_return* =true, and create a splice for each start and end of the break. To create new periods the original period should be copied for each splice and the following elements are modified:
   a) The presentation time of the splice can be used as the `Period`@start time of the ad period
   b) The `Period`@id may be set to the `Period`@start time as this value will be unique.
   c) When a `SegmentTimeline` is present, the segment timeline is used to find all segment presentation times and durations that overlap the period sufficiently (e.g. for 98 percent), then the corresponding `SegmentTimeline` with corresponding S elements in each Period, is rewritten. This way each Period only references sufficiently overlapping segments.

d) The `SegmentTemplate`@presentationTimeOffset shall be set to a time matching the event presentation time or Period@start, when converted to the correct timescale.

e) When an `EventStream` element is present, set the `EventStream`@presentationTimeOffset and iterate over each event, and remove events that do not overlap the period.

f) In case MPD@type='static' update the `Period`@duration accordingly.

g) Remove periods that have AdaptationSets with no segments.

This method of period splitting works for media presentation with MPD@type='static'. For MPD@type='dynamic' it is important that MPD@timeShiftBufferDepth is larger than the maximum interval between ad breaks such that SCTE 35 signalling can be used consistently to determine the @start of each Period.

5 SCTE 35 in Timed Metadata Track

The previous section discussed signaling SCTE 35 information in the manifest/ media presentation description. An alternative way considered for DVB-TA is to send the SCTE 35 metadata information directly in the media content segments rather than the manifest, DASH enables this by using `DASHEventMessageBox` and `InbandEventStream` element and the scheme_id_uri "urn:scte:scte35:2013:bin" as defined in SCTE 214 enabling direct encapsulation of the SCTE 35 data.

A third alternate approach described in this section is to store the data in separate track and specifically a timed metadata type of track. Recently ISO/IEC standardized 23001-18:2022 event message track format [16]. This format enables storing event messages in a separated track as an ISO Base media file format timed metadata track. This separate construct enables de-multiplexing of the `DASHEventMessageBox` information as described in approach two.

Using such a separate track format to send this data has several benefits. First, duplication in each track and/or each updated MPD in approaches one and two are avoided. Second, the storage and transport in a (fragmented) ISO base media file format [15] timed metadata track becomes consistent with the handling of media and timed text. Additionally, this information can be generated and stored separately from other media data.

This format [16] for the timed metadata track is very similar as storing webvtt in mp4, i.e. the boxes are stored in media samples. In ISO/IEC 23001-18 the `EventMessageInstanceBox` is defined with 4cc `emib`. It is stored in a sample and this box has a signed presentation time delta relative to the sample presentation time. For some samples that do not overlap an active marker, an `EventMessageEmptyBox` (embe) can be inserted in the sample. The rule for introducing sample boundaries is similar as in webvtt in MP4, where a boundary is created each time the set of active events changes. In other cases, such as a segment boundary, a sample boundary may be introduced as well.

A key benefit of using this third approach, is that a completely separate process can be used to generate the metadata track with SCTE 35 information, as there is no need for the encoder to multiplex the metadata with the segment or to update the MPD. The similarity with webvtt mp4 may also make it more trivial to implement this format in web browsers). In the Table 2 we show an example of mapping 5 Events with a presentation time, duration and id to event message instance boxes and event message track samples. As you see each sample boundary occurs when the active set of event messages changes. The `EventMessageEmptyBox` is indicates that no events are active during that sample.

Table 2 Example Events

Event@id	Presentation time	Event@duration
4	2	18
0	3	0
1	14	9
2	136	11
3	136	7
Sample		payload
1	0	EventMessageEmtpyBox
2	2	EventMessageInstanceBox(id=4,presentation_time_delta=0,event_duration=18)
3	3	EventMessageInstanceBox(id=0,presentation_time_delta=0,event_duration=0) EventMessageInstanceBox(id=4,presentation_time_delta=-1,event_duration=18)
4	4	EventMessageInstanceBox(id=4,presentation_time_delta=-2,event_duration=18)
5	14	EventMessageInstanceBox(id=1,presentation_time_delta=0,event_duration=9) EventMessageInstanceBox(id=4,presentation_time_delta=-12,event_duration=18)
6	20	EventMessageInstanceBox(id=1,presentation_time_delta=-6,event_duration=9)
7	23	EventMessageEmtpyBox

Table 3 Snippet of an event message track in the MPD

```
<AdaptationSet codecs="evte" contentType="meta"
mimeType="application/mp4" id="1"
startWithSAP="1"><SupplementalProperty
schemeIdUri="urn:dashif:events:metadataconfiguration:2022"
 value="urn:scte:scte35:2013:bin "/>
 <Representation bandwidth="8000" id="meta-track">
 <SegmentTemplate initialization="segments-$RepresentationID$.dash"
media="segments-$RepresentationID$-$Time$.dash" timescale="1000">
 <SegmentTimeline>
 <S d="2000" t="0" r="29"/>
 </SegmentTimeline>
</SegmentTemplate>
 </Representation>
</AdaptationSet>
```

In case multiple events are active during a sample, multiple `EventMessageInstanceBox`es are carried in the sample. In case a segment boundary is required, it is always possible to create a new sample with corresponding `EventMessageInstanceBox` es aligned with that time.

The event message track can be formatted as fragmented/mp4 data [15] or as Common Media Application Format data [17], in this case fragments can contain 1 or more samples of the track and utilization of similar transport for media and timed text is possible. To support signaling in the media presentation, the DASH-IF IOP version 5 part 10 [8] specified recommended signaling of the event message track. We provide an example snippet of a DASH Media presentation of this concept as described in Table 3. The timed metadata track is signaled in its own `AdaptationSet`, and the @codecs attribute is set to `evte`. The `AdaptationSet`@contentType is set to `meta` to indicate that this is a metadata track. To enable providing information about the contents a supplemental property that was defined by DASH-IF [9] can be used. In this case it was set to "urn:scte:scte35:2013:bin" the scheme for SCTE 35 event payload.

6 Proof of Concept Evaluations

6.1 Period Splitting

The first experimental evaluation shows how the signaling of SCTE 35 in MPEG DASH can be used to split periods with `SegmentTemplate` and `SegmentTimeline` from a single to a multi period MPD enabling manifest manipulation for ad insertion. First we implemented the procedure as outlined in clause 4.3 and implemented using Python and deploying it as an AWS lambda cloud function. Second we linked the script to an input: a live stream setup with several ad slots and content transitions signaled by SCTE 35 messages [21]. We then show some results obtained from the setup when playing the stream using a player (dash.js and shaka player) and applying the period splitting in real-time on-the-fly using the cloud function. The implementation is available on request to the author. Through monitoring of the function, when playing the on-the-fly multi-period mpd shows some performance results indicated in Figure 3. This performance results in 20 invocations per second with 100 percent success rate at an average 920 ms minimum, 845 ms maximum. The modified DASH MPD now contains 5-7 periods and plays continuously in different players we tested.

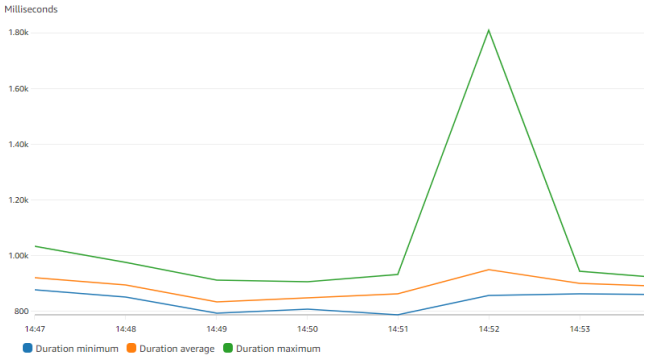

Figure 2 Results of request for multi period MPD

6.2 Generating Event Message Tracks

To support the adoption of the event message track format, an example implementation was created and made available on github under the link in [20]. The repository contains different sample programs to experiment with this format. *Generate_example* prints the examples of mapping (random) events to even message track samples as shown in Table 2. The program *fmp4_dash_event* and *dash_event_fmp4* enable converting metadata in fmp4 to XML and vice versa. *print_event_samples* was created for inspecting the different event track samples and *gen_avail_track* enables generating a track with SCTE 35 ad slots with *splice_insert* markers periodically inserted, allowing the specify media segment duration, avail duration, avail interval duration and track duration. The implementation makes it easy to generate some sample test files.

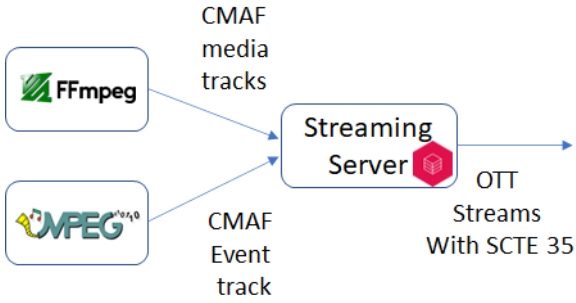

Figure 3 Distributed uplink with CMAF live ingest

6.3 Distributed uplink Live uplink

In Figure 3 we show another setup of the event message track, in this case the ABR encoder is implemented using FFmpeg encoder[24] and it is transmitting media segments to a streaming server based on unified origin. In this case we implemented a separate routine push markers [22] as available that can generate markers using the tools developed in the previous section and push them as CMAF [17] media segments with regular durations. By aligning the segments and timeline origin between the FFMPEG and [22], we are able to implement a setup with distributed encoding and distributed ingest of timed metadata and splice point information. By deploying the distributed setup in a Kubernetes cluster, we upkeep a robust DASH live stream with SCTE 35 [23].

7 Discussion and Conclusion

In the first part of this paper, we presented the reference architecture for dynamic ad insertion in OTT DASH workflows and the relevance of SCTE 35 for signaling ad slot opportunities. From the discussion between DVB and SCTE, this paper summarizes and presents the constraints and interpretations codified into either DVB-TA part 3 or SCTE 214 to enable maximum compatibility for server and player/client-side ad insertion. The main principle is that the event timing predicts the upcoming the splice point. In addition, a new ad slot signaling was introduced for insertion for video on demand content and lastly the early termination case was discussed. As proof of some of these concepts, we implemented a simple stateless cloud function that splits periods of the original input MPD in multiple periods using SCTE signals. In the second part of the paper SCTE 35 markers in timed metadata tracks are discussed and some sample implementation is provided to help implementers. In a next update we will provide more detail on corresponding HTTP Live Streaming signalling.

ACKNOWLEDGMENTS

We thank all contributors in DVB, SCTE, DASH-IF and MPEG for their input and viewpoints on the respective specifications discussed in this document.

REFERENCES

[1] ANSI/SCTE 35 Digital Program Insertion Cueing Message available: https://www.scte.org/documents/5408/SCTE_35_2022b.pdf, 2022

[2] ANSI/SCTE 214-1 MPEG DASH for IP-Based Cable Services Part 1: MPD Constraints and Extensions, 2022 Available : https ://www.scte.org/documents/5147/ANSI_SCTE_214-1_2022.pdf

[3] ISO/IEC 23009-1:2022 Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats Available:https://standards.iso.org/ittf/PubliclyAvailableStandards/c083314_ISO_IEC%2023009-1_2022(en).zip

[4] Internet Engineering TaskForce (IETF) R.Pantos, RFC 8216 HTTP Live Streaming. Available: https://www.rfc-editor.org/rfc/rfc8216 [3]

[5] ANSI/SCTE 250 2022 Real-time Event Signaling and Management API Available https://www.scte.org/documents/5635/ANSI_SCTE_250_2022.pdf, 2022

[6] ANSI/SCTE 130-9 2020 Recommended Practices for SCTE 130 Digital Program Insertion—Advertising Systems Interfaces, 2020

[7] ANSI/SCTE 67 2017 *Recommended Practice for Digital Program Insertion for Cable,* 2017 Available: https://www.scte.org/documents/265/ANSI_SCTE-67-2017-1576856953355.pdf

[8] DASH-IF, DASH-IF IOP guidelines version 5 part 5: Ad insertion. 2021
Available : https ://dash-industry-forum.github.io/docs/IOP-Guidelines/DASH-IF-IOP-Part5-v5.0.0.pdf

[9] DASH-IF, DASH-IF IOP version 5 Part 10: Events and Timed Metadata, 2023
Available : https ://dash-industry-forum.github.io/docs/IOP-Guidelines/DASH-IF-IOP-Part10-v5.0.0.pdf

[10]SNPTV AFMM Addressable TV Guidelines
Version UK 2.0.6 – SNPTV -AFMM
Available : https ://www.snptv.org/wp-content/uploads/2020/08/SNPTV-AFMM-Addressable-TV-UK-version-2.0.6.1.pdf

[11] Interactive Advertising Bureau, Digital Video Ad Serving Template (VAST) version 4.3
Available: https://iabtechlab.com/wp-content/uploads/2022/09/VAST_4.3.pdf

[12] European Telecommunication and Standardization Institute (ETSI) TS 103 752-1: DVB-TA (Targeted Advertising – Part 1: broadcast signalling) Dynamic substitution of content in linear broadcast – Part 1: carriage and signalling of placement opportunity information in DVB Transport Streams
Available : https://www.etsi.org/deliver/etsi_ts/103700_103799/10375201/01.01.01_60/ts_10375201v010101p.pdf

[13] Digital Video broadcasting (DVB): DVB-TA (Targeted Advertising – Part 3: DASH signalling) Dynamic substitution of content in linear broadcast – Part 3: carriage and signalling of placement opportunity information in DVB-DASH Interim bluebook draft
Available: https://dvb.org/wp-content/uploads/2022/08/A178-3_Dynamic-substitution-of-content-in-linear-broadcast_Part3_Signalling-in-DVB-DASH_Interim_Draft-TS-103-752-3v111_Aug-2022.pdf

[14]Media Perspectives. *Event Triggering Distribution Specification (ETDS)* available: https://mediaperspectives.nl/app/uploads/2018/10/2018-10-16-Media-Perspectives-ETDS.pdf , October 2018

[15] ISO/IEC 14496-12:2022 Information technology — Coding of audio-visual objects — Part 12: ISO base media file format Available : https://www.iso.org/standard/83102.html

[16]ISO/IEC 23001-18:2022 Information technology — MPEG systems technologies — Part 18: Event message track format for the ISO base media file format
Available : https ://www.iso.org/standard/82529.html

[17] ISO/IEC 23000-19:2020 Information technology — Multimedia application format (MPEG-A) — Part 19: Common media application format (CMAF) for segmented media

[18] R. Seeliger, D. Silhavy and S. Arbanowski, "Dynamic ad-insertion and content orchestration workflows through manifest manipulation in HLS and MPEG-DASH," *2017 IEEE Conference on Communications and Network Security (CNS)*, Las Vegas, NV, USA, 2017, pp. 450-455, doi: 10.1109/CNS.2017.8228708

[19] R. Seeliger, L. Bassbouss, S. Arbanowski and S. Steglich, "Towards Personalized Content Replacements in Hybrid Broadcast Broadband Environments," *2019 23rd International Computer Science and Engineering Conference (ICSEC)*, Phuket, Thailand, 2019, pp. 385-389, doi: 10.1109/ICSEC47112.2019.8974706.

[20] Online: Event Message Track Sample implementation: unifiedstreaming/fmp4-ingest: Repository on shared work on developing a fragmented MPEG-4 ingest specification (github.com)

[21] Online: SCTE 35 sample stream available: https://demo.unified-streaming.com/k8s/vod2live/stable/unified-learning.isml/.mpd last accessed April 7 2023

[22] Online: fmp4ingest tools: https://github.com/unifiedstreaming/fmp4-ingest last accessed April 7 2023

[23] Online: SCTE 35 sample live stream available: https://demo.unified-streaming.com/k8s/live/stable/scte35.isml/.mpd last accessed April 7 2023

[24] Online: FFmpeg encoder: https://ffmpeg.org/ last accessed April 7 2023