CHANGE REQUEST

DASH-IF IOP **CR** **rev** **-** **Current version:** V4.3

Status:		Draft		Internal Review	X	Community Review		Agreed

Title:	Low-latency Modes for DASH
Source:	Live TF
Supporting Companies:	Akamai, castLabs, Comcast, Elemental Technologies, Ericsson, Harmonic, Qualcomm Incorporated, Sony, TNO, Unified Streaming, Frontier Communications
Category:	**A** ***Date:*** 2019-12-21

Use one of the following categories:
C *(correction)*
A *(addition of feature)*
B *(editorial modification)*

Reason for change:	DASH-IF collected information related to Low-Latency Streaming in a Report, together with DVB. The report (available here: https://dash-industry-forum.github.io/docs/Report%20on%20Low%20Latency%20DASH.pdf) provides use cases, service scenarios, deployment experience and existing potential technologies. Also, DASH-IF already generated the DASH profile for ATSC that includes a mode supporting low latency. Based on this information in the report, it is recommended to add low latency to DASH-IF IOP Guidelines.
Summary of change:	This change provides a new clause for live services that addresses specification updates as well as implementation guidelines to support Low-Latency DASH services addressing the requirements above.
Consequences if not approved:	Low-Latency DASH not available

Sections affected:	References in clause 2 New section 9.X in v5 of DASH-IF IOP
Other comments:	This document contains several notes highlighted in yellow. Feedback during community review is welcome specifically on these topics.

Disclaimer:	This document is not yet final. It is provided for public review until the deadline mentioned below. If you have comments on the document, please submit comments by one of the following means: - at the github repository https://github.com/Dash-Industry-Forum/Live/issues (public at https://gitreports.com/issue/haudiobe/Live), or - dashif+iop@groupspaces.com with a subject tag [LL-DASH] Please add a detailed description of the problem and the comment. Based on the received comments a final document will be published latest by the expected publication date below, integrated in a new version of DASH-IF IOP if the following additional criteria are fulfilled: - All comments from community review are addressed - The relevant aspects for the Conformance Software are provided - Verified IOP test vectors are provided
Commenting Deadline:	January 31st, 2020
Expected Publication:	March 31st, 2020

References

[X1] DASH-IF IMPLEMENTATION GUIDELINES: TOKEN-BASED ACCESS CONTROL FOR DASH (TAC) V1.0: http://dashif.org/wp-content/uploads/2016/07/DASH-TAC-v0.9.pdf

[X2] ISO/IEC 23009-1:2020: Draft of ISO/IEC 23009-1:2014 4th edition (available in w18416)

[X3] DASH-IF/DVB Report on Low-Latency Live Service with DASH, July 2017, available here: https://dash-industry-forum.github.io/docs/Report%20on%20Low%20Latency%20DASH.pdf

[X4] DASH Ingest Protocol, available for community review here: https://dashif-documents.azurewebsites.net/Ingest/master/DASH-IF-Ingest.html

[X5] CTA-5003: "CTA WAVE Device Playback Capabilities Specification (CTA-5003)", available here: https://cta.tech/cta/media/EventImages/TechStandards/CTA-5003-Final.pdf

[X6] FDIS of ISO/IEC 23009-1 4th edition http://wg11.sc29.org/doc_end_user/documents/127_Gothenburg/wg11/w18609.zip

[X7] DVB-DASH BlueBook A168, "MPEG-DASH Profile for Transport of ISO BMFF Based DVB Services over IP Based Networks", October 2019, http://dvb.org/wp-content/uploads/2019/12/a168_dvb_mpeg-dash_oct_2019.pdf

[X8] CDAM ISO/IEC 23009-1 4th edition CDAM 1 CMAF support, events processing model and other extensions, http://wg11.sc29.org/doc_end_user/documents/128_Geneva/wg11/w18926.zip.

9 DASH Live Services

9.X Low Latency DASH

9.X.1 Introduction

Based on a report developed jointly between DVB and DASH-IF on Low-Latency DASH [X3], this clause defines details on how to support consistent latency in DASH for linear TV services. The Low Latency DASH scenario is a variant of the Live Services recommended approach focused on ensuring that the end-to-end latency of the streams will be comparable to the broadcast latency. This broadcast latency is not a universal value, as it is influenced by many factors such as the duration of the broadcast encoding pipeline, the latency of the transport channel which can slightly differ per type (satellite, cable, IPTV, DTT...), or the artificial delays introduced by local content moderation regulations. However,

most of the measurements converge on a 3 - 10 seconds latency between the moment where the source signal is acquired for encoding and the moment when it's played back on the TVs. For details refer to [X3].

9.X.2 Scenario and Architecture

A typical workflow presented in Figure 1. This figure does not present a mandatory deployment architecture, but is used to illustrate the considered interfaces addressed in this clause. The source signal is encoded in several resolutions/bitrates by the ABR encoder, which produces fragmented MP4 or MP4/CMAF compliant elementary streams and eventually encrypts them. Those streams are then ingested into the packager using a protocol suitable protocol to deliver the data in streams/chunks. The packager proceeds with the necessary operations such as the transformation of the incoming streams' segments duration, DRM preparation of the streams, and the generation of the DASH manifests.

Basically, two ways are defined to generate low-latency Representations and Media Presentations.

In the first case, Segments short enough to address the latency are generated. In a second more advanced way, the resulting set of files is made available on a disk or RAM storage in such a way that the origin server can load the chunks while they are packaged by the packager.

In parallel, the packager needs to generate the MPD, and the DASH clients are requesting the DASH MPD and leverage it to calculate the live edge time of the stream. Whereas in the first case, regular DASH client and Low-Latency Client operate similar, in the second case with chunking, the following applies:

- when a regular DASH Client does the calculation, it defines which Segment is supposed to be completely available on the Origin Server and requests it from the CDN. Every media segment is fully fetched by the Client before being added to the playback buffer.

- when a Low Latency DASH Client does the calculation, it should do the same calculation but request the next segment when its first chunk becomes available on the origin, signaled by the `@availabilityTimeOffset` parameter from the DASH manifest. Every media segment can be added to the playback buffer in one or several steps, as the CDN is delivering the Live Edge Segment in HTTP chunked transfer encoding mode.

Regular DASH client schedules the playback according to their own timeline. Low Latency clients follow a service description that is provided in the MPD or by external means. The decryption and decoding processes are identical for both Client types, with the difference that their scope can be a full segment or a partial segment.

When the request for a DASH manifest comes to the CDN, it can leverage an intermediate manifest manipulation service instead of requesting the manifest straight from the Origin Server. This covers use cases where the manifest has to be customized for each end-user or group of end-users, with such manipulations as ad insertion, bitrate filtering, A/B watermarking or content replacement for blackout use cses. The manifest manipulation operations is not expected to significantly impact the end-to-end latency, at least not more than the duration of a single media segment.

Requests for media segments are all forwarded to the Origin Server. Ideally, all requests for all stream components (manifests and media segments) should be done in HTTPS on the DASH Client side, and forwarded in HTTPS to all downstream services (e.g., Manifest manipulation), in order to avoid protocol downgrade from an initial request in HTTPS to downstream requests in plain HTTP. In order to protect DRM services, all requests should be sent by the DASH Client to the CDN, possibly using a tokenization mechanism such as the one defined in the TAC specification [XX], so that no illegitimate request reaches the DRM License server during the playback session.

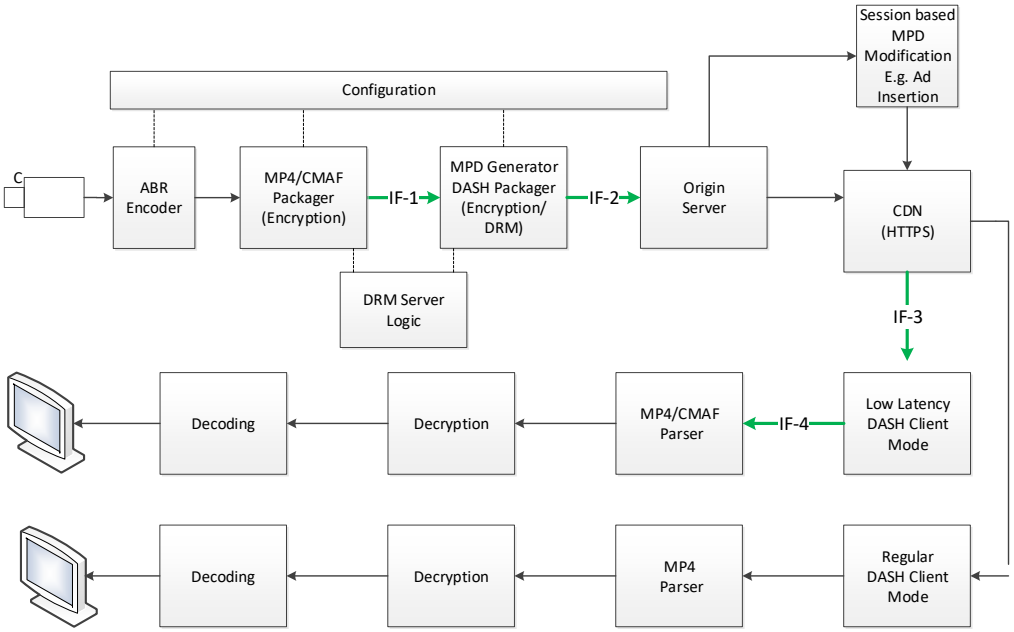

Figure 1: Typical DASH-IF Low-Latency Architecture

When chunked MP4/CMAF is used in the Low Latency DASH scenario to enable such latency levels to be reached, the stream structure should be backward compatible with DASH players that rely on the reception of a full media segment before the playback of the stream can start. But HTTP chunked transfer encoding shall at least be supported up from the ingest into the packager up to the CDN edge, whereas the last mile delivery shall happen using HTTP chunked transfer encoding or HTTP in regular mode. If HTTP chunked transfer encoding is supported by the DASH player, it basically means that a media segment carrying the latest moment of the program (also known as the "live edge time") could be consumed on the player while it's still being produced by the encoder and the packager.

Figure 2 provides a basic flow of information for operating a low-latency DASH service. The DASH packager gets information on the general description of the service as well as the encoder configuration. The encoder produces CMAF chunks and fragments. The chunks are mapped by the MPD packager onto Segments and provided to the network in incremental fashion. Segments are not delivered as a whole, but progressively as they are generated.

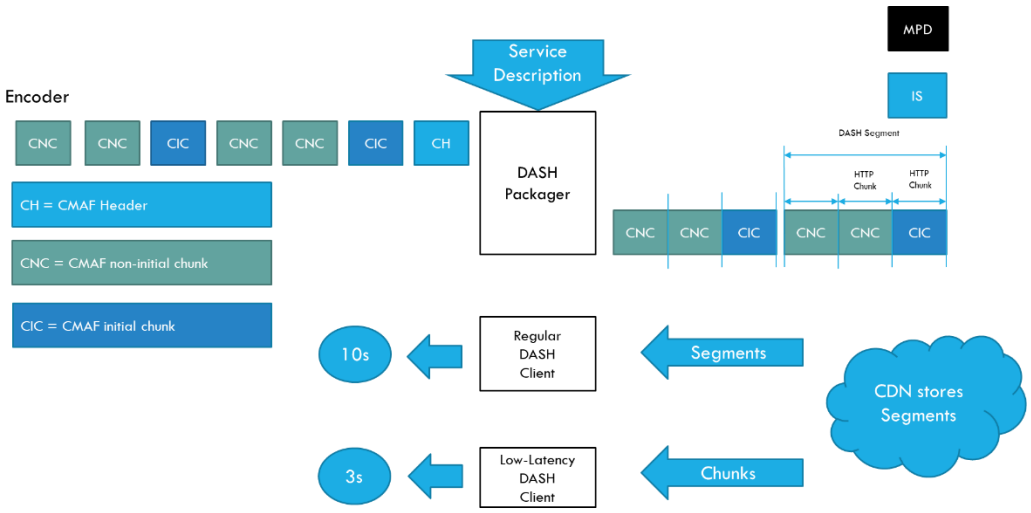

Figure 1 Basic operation flow Low-Latency DASH

The key technologies for LL-DASH are briefly summarized in the following. Detailed requirements for service offering and clients are provided in clause 9.X.4 and 9.X.5, respectively.

1. The encoding is based on fragmented ISO BMFF files and CMAF Fragments and CMAF chunks are assumed.

2. Each chunk is individually accessible by the DASH packager and gets mapped to an HTTP chunk that is uploaded to an origin server. This 1-to-1 mapping is a recommendation for low-latency operation, but not a requirement. By no means, the client should assume that this 1-to-1 mapping is preserved to the client.
3. A low delay protocol, e.g. HTTP Chunked Transfer Encoding, of partially available Segments is used such that clients can access the Segments before they are completed. The availability start time is adjusted for clients being able to make use of this feature.
4. Two operation modes are permitted
    - Simple live offering is used by applying `@duration` signaling and `$Number$` based templating
    - Main live offering with the **`SegmentTimeline`** as either `$Number$` or `$Time$` is supported by the proposed updates in MPEG-DASH fourth edition.
5. MPD validity expiration events are encouraged to be used but are not essential to be understood by the clients.
6. Generally, Inband event messages may be present, but clients are only expected to recover those at the start of Segments, not at arbitrary chunks. DASH packagers however may receive notifications from the encoder at chunk boundaries or completely asynchronuously using timed metadata tracks. For details refer to the DASH-IF ingest protocol [X4].
7. It is permitted in a single Media Presentation and within one Period of a Media Presentation to have Adaptation Sets that are using the chunked low-latency mode and Adaptation Sets using short Segments for different media types.
8. A certain amount of playback control of DASH client on the media pipeline may be available and should be used for robustness of DASH clients, for example the playback may be accelerated or decelerated for some period of time or may do a seek into Segments.
9. The system is designed to be workable with standard HTTP/1.1, but also should be applicable to HTTP extensions and other protocols for improved low-latency operation.
10. The MPD includes explicit signaling on the service configuration as well as the service properties (for example including the target latency of the service).
11. MPDs and possibly also Segments include anchor times that allow DASH clients to measure the current latency compared to live and adjust to meet the the service expectations.
12. The MPD contains information on the chunk structure as well as on possibly availably additional random access and switch points,
13. Operational robustness is addressed, for example in case of encoder failures.
14. Existing DRM and encryption modes are compatible with proposed low-latency operation.

This specification aligns with the DVB-DASH specification (including the low-latency extensions) [X7] and joint development and design meetings were held between DVB and DASH-IF to ensure industry alignment. Any differences to the DVB DASH low latency specification are pointed out.

A brief summary of how this extended DASH specification for low-latency addresses some of the issues is provided in the following:

1. This specification extension is fully aligned with earlier DASH specifications and basically no changes are done to core MPD and Segments formats. This specification only provides sufficient guidelines for a service provider and client implementor to make use of additional functionalities and signaling to support low-latency. Legacy clients are expected to still be able to use the service, just at some higher latency.
2. This specification makes full use of the ubiquitously available HTTP/1.1, and modern CDN architectures are therefore fully usable, possibly with some updates on handling the relay of partially complete Segments through the network. This specification recommends the use of HTTP Chunked Transfer encoding for reduced latency which is a mandatory client part of HTTP/1.1. Secondly, the low-latency content is fully cacheable for consumption in time-shift and network-PVR mode without modification.
3. This specification enables CDN-cache-friendly and efficient operation as the Segments that are stored on the CDN for later consumption are of typical Segment duration ranges of several seconds.
4. This specification may benefit from improved network protocols, such as HTTP/2 or HTTP/3, but does not require any of those. Details are for further study.
5. This specification relies on existing MPD compactness based on segment templates using the same approach as for regular DASH live services, i.e., no changes on MPD processing and generation are required.
6. This specification relies on the existing DASH timing model and asynchronous client-server time synchronization. By this, a DASH client once synced to the server has full knowledge, when a Segment was published and when the next segment will be published without requiring more frequent MPD updates than done for a regular DASH service. This knowledge can be used be the DASH client to request the proper Segment at startup and at the right time (i.e., when it becomes available) and also maintain consistent end-to-end latency.

7. This specification also supports fast random access without any MPD changes. Random access in the middle of Segments is indicated in the MPD as well as in Segments. In addition, fast joining may be supported by the combination of knowledge on segment availability and accelerated decoding.
8. The specification fully relies on technologies defined in MPEG in updates for ISO/IEC 23009-1 (MPEG-DASH) and ISO/IEC 23000-19 (CMAF).

In the same timeline as this specification was developed and published, another CMAF based streaming specification[1] was published to address low-latency and was analyzed here[2], raising some aspects discussed above.

9.X.3 Definitions

Low-Latency Service Offering: A Service Offering that contains information to support a Low-Latency DASH client to provide the service in the target latency of typically 2 to 10 and the target latency can be defined by the service provider.

Low-Latency Adaptation Set: An Adaptation Set that can be consumed based on the low-latency service offering.

Low-Latency Chunked Adaptation Set: A Low-Latency Adaptation Set that is offered containing Representations that follow the requirements of this specification, namely providing early access to Segments as well as CMAF chunked content.

Low-Latency Segmented Adaptation Set: A Low-Latency Adaptation Set that is offered containing Representations that follow the requirements of this specification, namely providing short enough Segments to be consumed as low-latency service.

Low-Latency DASH Client: A client that follows this specification in order to consume the Media Presentation offered as Low-Latency Service Offering and can meet the target latency requirements.

9.X.4 Low-Latency Service Offering

9.X.4.1 Introduction

A Media Presentation that follows a DASH-IF Low-Latency Service Offering according to this specification should be signalled with the `@profiles` identifier `'http://www.dashif.org/guidelines/low-latency-live-v5'`.

If an MPD signals conformance to this profile, it shall follow the requirements and recommendation of the specification in clause 9.X.4.2 and the referenced clauses.

A legacy service offering is documented in clause 9.X.3.6. This legacy offering does not conform to the DASH-IF Low-Latency Service Offering profile but may used in practice.

DVB Low Latency DASH is documented in TS 103 285 [X7]. Compatibility aspects are documented in clause 9.X.4.7.

9.X.4.2 General Service Offering Requirements and Recommendations

A Low-Latency Service Offering follows the following requirements and recommendations:

1. At least one **`ServiceDescription`** element shall be present as follows:
    - **`Scope`**: Specifies the scope of the service description element. If present, this service description only targets DASH clients within the scope of this descriptor. DASH clients not in scope, i.e., not recognizing any of the scope descriptor elements, are expected to ignore this service description. The scope may for example be used offering content to clients on different network conditions for which the expectation of the target latency is different. DASH-IF does not define a scope description at this stage, it focuses on the interpretation of the Service Description assuming that the client is in scope.
    - A **`Latency`** element shall be present including a

---

[1] https://developer.apple.com/documentation/http_live_streaming/protocol_extension_for_low-latency_hls_preliminary_specification

[2] https://www.linkedin.com/pulse/apples-implementation-low-latency-hls-explained-phil-harrison/

        - `@target`: The service provider's preferred presentation latency in milliseconds compared to the producer reference time.

      The **`Latency`** element may also contain one of the following parameters:

        - `@max`: The service provider's indication about the maximum presentation latency in milliseconds. Indicates a content provider's desire for the content not to be presented if the latency exceeds the maximum latency.
        - `@min`: The service provider's indication about minimum presentation latency in milliseconds for example to avoid inconsistencies with second screen applications, overlays, etc.

    - A **`PlaybackSpeed`** element may present to providea permission on the boundaries for this service. If present, it shall contain at least
        - `@max`: The maximum playback rate that the content provider indicates is appropriate for the purposes of automatically adjusting playback latency and buffer occupancy during normal playback, where 1.0 is normal playback speed.
        - `@min`: The minimum playback rate that the content provider indicates is appropriate for the purposes of automatically adjusting playback latency and buffer occupancy during normal playback, where 1.0 is normal playback speed.
    - Other service description parameters may be present, for details refer Annex K of the ISO/IEC 23009-1, 4th edition [X6].

2. At least one UTC timing description shall be present and be restricted to one of the followings `@schemeIdUri` set to one of the following:
    - `urn:mpeg:dash:utc:http-xsdate:2014`
    - `urn:mpeg:dash:utc:http-iso:2014`
    - `urn:mpeg:dash:utc:http-ntp:2014`

3. Low latency content should provide a **`LeapSecondInformation`** element providing correction for leap seconds. When the timing of a future leap second becomes known, the **`LeapSecondInformation`** element should be updated to include `@nextLeapChangeTime` and `@nextAvailabilityStartLeapOffset` attributes with sufficient notice for players to receive the information before it is needed. Such an update should not be made less than 24 hours after a previous leap second has occurred to ensure that players have time to process the previous leap second. For more details, refer to clause 9.X.6.2.10.

4. Inband Event Streams carrying MPD validity expiration events as defined in clause 4.5 should be present. If used, `@value` shall be set to 1.

5. For each media type at least one Low-Latency Adaptation Set shall be present as defined in clause 9.X.4.3. This includes that only Low Latency Segment Adaptation Sets as defined in 9.X.4.4 or only Low Latency Chunked Adaptation Sets as defined in 9.X.4.5 or a combination of both may be present.

6. Event Message Streams may be used in low-latency media presentations. If Inband Event Streams are present, then they should be carried in Low Latency Segment Adaptation Sets as defined in 9.X.4.4., if present.

7. Operational robustness is expected to be provided by the means documented in clause 4.8.

8. For encrypted content, the rules documented in clause 7 apply. Additional guidelines are provided below.

Service Implementation guidelines are provided in clause 9.X.6.

9.X.4.3 Low Latency Adaptation Set

A Low-Latency Adaptation Set follows the following requirements and recommendations:

1. It shall include at least one **`ProducerReferenceTime`** element with the following constraints
    - `@id`: provide a unique id for this reference
    - `@type`: shall be set either to `encoder` or `captured`
    - A **`UTCTiming`** element that is identical to one present in the MPD shall be present and is used for deriving the value of the `@wallclockTime`.
    - `@wallclockTime`: shall be present and provide the value at the `@presentationTime`

- `@presentationTime`: shall be the value of the `@presentationTimeOffset`, if present, or 0 otherwise. Note that this means that the producer reference time is provided at the start of the Period.

2. A Low-Latency Adaptation may either be a Low-Latency Segment Adaptation Set as defined defined in clause 9.X.3.4 or a Low-Latency Chunked Adaptation Set as defined in clause 9.X.3.5.

3. One of the two shall be present:
   - the **`SegmentTemplate`**`@duration` attribute and the **`SegmentTemplate`**`@media` with `$Number$` or
   - the **`SegmentTimeline`** element and the **`SegmentTemplate`**`@media` with `$Number$` and `$Time$`

4. Segment durations for one Representation should not vary significantly. In particular, the segment duration shall not vary more than indicated in the MPD applying the rules defined in clause 4.11.3.7.

9.X.4.4 Low Latency Segment Adaptation Set

A Low-Latency Segment Adaptation Set follows the following requirements and recommendations:

1. It shall conform to a Low-Latency Adaptation Set as defined in clause 9.X.4.3.

2. Each Segment should include only a single movie fragment header `'moof'`. If they include only a single `'moof'`, then the Segment may carry a `'smds'` brand and shall signal this by providing the `@segmentProfiles` being including the `'smds'` brand.

3. the **`SegmentBase`**`@availabilityTimeComplete` shall be absent, indicating that the Segments are completely available when requested.

4. The Segment duration shall not exceed 50% of the value of the target latency and should not exceed 30% of the target latency.

Low Latency Segment Adaptation Sets are identified by the setting in 2 and 4 above as well as by the absence of the **`Resync`** element.

9.X.4.5 Low Latency Chunked Adaptation Set

A Low-Latency Chunked Adaptation Set follows the following requirements and recommendations:

1. It shall conform to a Low-Latency Adaptation Set as defined in clause 9.X.4.3.

2. Each Adaptation Set shall conform to an Adaptation Set according to the DASH profile for CMAF content as defined in MPEG DASH, clause 8.X.4.

3. Each Segment shall conform to a CMAF Fragment but may and typically should contain more than one CMAF chunk. CMAF chunks should be generated such that the range of presentation times contained in any CMAF chunk of the CMAF Track do not overlap with the range of presentation times in any other CMAF chunk of the same CMAF Track.

4. A **`Resync`** element should be assigned to each Representation (possibly defaulted) signaling the properties of the Segments and the used chunks (chunk size, chunk duration, chunk properties). For details on the usage of the **`Resync`** element, refer to clause 9.X.6.2.8.

5. the **`SegmentBase`**`@availabilityTimeOffset` shall be present on Adaptation Set level andshall be

   - greater than zero,
   - smaller than the maximum segment duration for this Representation,
   - such that the difference of the value of the **`SegmentBase`**`@availabilityTimeOffset` and the maximum segment duration for this Representation is smaller than the target latency.

6. the **SegmentBase**@availabilityTimeComplete shall be present on Adaptation Set level and set to 'FALSE'.

7. Any 'emsg' box shall be placed as follows:
    - It may be placed in between any 'mdat' and 'moof' box or before the first 'moof' box
    - If not placed before the first 'moof' box, an equivalent 'emsg' with the same id value shall be present before the first 'moof' box of the next Segment.

8. One of the following should be applied for any Adaptation Set that contains more than one Representation:

    - the segment duration is smaller than the anticipated client buffer size. An upper bound to the client buffer size is the target latency as provided in the service description. Hence, the maximum segment duration shall be smaller than the signaled target latency and should be smaller than half of the signaled target latency, or

    - at least one Representation is present with @bandwidth value is the lowest in the Adaptation Set and the and it contains a **Resync** element with the following settings
        - the @type value is set to 1 or 2.
        - The @dT is set such that the value normalized by the @timescale of the Representation is at most the signaled target latency and should be smaller than half of the signaled target latency.
        - @marker is set to TRUE.

      The @qualityRanking should be used for this Representation with an indication that quality is worse than the one of all other Representation.

      If this mode is used, then in additional the following applies for all Representations in Adaptation Set:
        - CMAF chunks shall be generated such that the range of presentation times contained in any CMAF chunk of the CMAF Track do not overlap with the range of presentation times in any other CMAF chunk of the same CMAF Track.
        - CMAF chunks shall be aligned in decode and presentation time across all Representations.

      Additional Representations with higher values for @bandwidth may be present with **Resync** set as above and permitting faster downswitching or faster random access. In the latter case the @type value may be set also 3. The @qualityRanking should be used.

    Note 1: This permits downswitching within Segments and Resynchronization within Segments. For details on content generation, refer to clause 9.X.6.2.8.

    Note 2: Other options to support faster downswitching exist. As an example, an Adaptation Set with the same content may be provided with smaller Segment duration. The signalling for such an Adaptation Set is for further study.

    Note 3: As the interpretation of the Resync is not guaranteed by legacy DASH clients including for example DVB LL-DASH clients, content authors should be careful in relying on this information.

9.X.4.6 Legacy Setup

As low-latency DASH services have been developed over time, this clause documents legacy aspects for DASH Low-Latency services compared to the DASH-IF Low Latency Service Offering profile.

In such a legacy setup one or more of the following applies:

1. The **ServiceDescription** element may not be present. In particular, the target latency may be absence, but may be provided by external means, for example by the application by setting the value through an API.
2. The **ProducerReferenceTime** element may is not present. In this case, the Period Start time is assumed to be used as the wall-clock time and the value of the @presentationTimeOffset is assumed as the corresponding presentation time.
3. The Segment durations for Low Latency Segment Adaptation Sets may exceed the 50% target value

4. The Low Latency Chunked Adaptation Sets may not conform to an Adaptation Set according to the DASH profile for CMAF content as defined in MPEG DASH.
5. The **Resync** element may not be present.

9.X.4.7 DVB Low-Latency DASH

DVB Low Latency DASH is documented in TS 103 285 [X]. As the specification was completed prior to the DASH-IF specification, a few issues important to be considered are documented in the following:

1. DVB-DASH in clause 11.18.2 recommends the use of `@duration` together with `$Number$`, whereas DASH-IF does not make any of such recommendations.
2. DVB DASH does not mention the **Resync** element, but the presence of the element would not break a DVB-DASH client.
3. DVB DASH signals Adaptation Sets that are for low latency purposes with an **EssentialProperty** or **SupplementalProperty** descriptor with the `@schemeIdUri` attribute of `"urn:dvb:dash:lowlatency:critical:2019"` and the `@value` attribute set to `"true"`. If the **SupplementalProperty** descriptor is used, then this does not impact regular low-latency DASH clients.

9.X.5 Low-Latency Client

For DASH-IF low-latency clients the following applies:

1. They shall be able to consume content offered with the DASH-IF Low Latency Service offering profile as defined in clause 9.X.4.2. Specifically, this includes
    - support media segments that contain more than one pair of `'moof'` and `'mdat'` boxes, where each moof/mdat pair may contain any number of ISO BMFF samples between 1 and the full segment duration inclusive.
    - Support playing two or more Adaptation Sets for which the Segments of one Adaptation Set do not align with the segments of another (e.g., due to differing segment durations) or the segments of one Adaptation Set contain multiple moof/mdat pairs and the segments of another only have a single moof/mdat pair. In particular playing back Low Latency Segment Adaptation Sets and Low Latency Chunked Adaptation Sets at the same time shall be supported.
2. They should be able to consume legacy low-latency content as defined in clause 9.X.4.6.
3. They should be able to consume DVB DASH low-latency content as defined in clause 9.X.4.7. For detailed requirements on DVB-DASH clients, please refer to TS 103 285, clause 10.20.

In the following, more detailed requirements and recommendations are provided, assuming a DASH-IF Low Latency Service offering profile. Indications when using consume legacy low-latency content are also provided.

4. For DASH-IF low-latency clients shall compute the presentation latency within each Period based on a wall clock anchor `WCA` and a presentation time anchor `PTA`. `WCA` and `PTA` are determined as follows:
    a. If the **ProducerReferenceTime** element is present as defined in clause 9.X.4.2, then the
        i. `WCA` is the value of the `@wallClockTime` in a format as specified by depending on the scheme of any **UTCTiming** element contained in this **ProducerReferenceTime**.
        ii. `PTA` is the value of the `@presentationTime` contained in this **ProducerReferenceTime** minus the value of the `@presentationTimeOffset` of the corresponding Representation.
        iii. `TS` is the `@timescale` of the corresponding Representation.
        iv. If the `@inband` attribute is set to `TRUE`, then the client should parse the Segments to re-verify the difference of `PTA/TS` and `WCA` based on the information in a `'prft'` box included in Segments.
    b. Else
        i. `WCA` is the value of the `PeriodStart`, i.e., the sum of **MPD**`@availailityStartTime` and **Period**`@start`,
        ii. `PTA` is the value of the `@presentationTimeOffset`
    c. Then the presentation latency PL of a presentation time PT presented at wall clock time WC in seconds is determined as `PL = (WC – WCA) – (PT – PTA)/TS`.

        Example: Let the `@timescale` be set to 20, and let the `@presentationTimeOffset` be set to 1000. Assume the presentation time of a sample being `4740` and the presentation time is

presented at `20:18:10.5` and the anchor of the wall-clock time is `20:15:00`. Then the presentation latency of the sample is derived as `190.5s - (4740-1000)/20 s = 190.5s - 187s = 3.5s`. Note that the format of the wall-clock time needs to follow the permitted formats in this specification.

5. It shall implement means to support the service description functionality (either in the DASH client signaled through MPD or by the means of an API, precedence is implementation specific). The parameter interpretation is as follows:
    a. The client when consuming in live mode, it should play the content within 500ms tolerance of the target latency taking into account the above computation for latency. However, the client should consider meeting the latency target also by taking into acccount knowledge of its own capabilities, the network conditions, and any relevant knowledge of past streaming performance. DASH-IF low latency clients should implement the Presentation time target and constraints in clause 10.20.4 of the DVB-DASH specification.
    b. If the max latency is set, the service provider indicates the maximum presentation latency in milliseconds. This value indicates a content provider's desire for the content not to be presented if the latency exceeds the maximum latency. A client when consuming in live mode, should not exceed the described max latency and, if it happens, inform the application that this event happens. Examples to maintain the latency may be to switch to lower available bitrates, drop certain media components, or other means to maintain the latency, to the extreme to even terminate the service.
    c. If the min latency is set, the client shall not fall below the described min latency.
    d. If the playback speed element is present, clients should use these tools to adapt to the target latency but should not exceed the max/min playback rates. DASH-IF low latency clients should implement the catch-up modes in clause 10.20.6 of the DVB-DASH specification.

6. Clients shall parse the `@availabilityTimeOffset` and should access Segments at adjusted segment availability times (i.e., compensated by `@availabilityTimeOffset`). Clients shall parse the `@availabilityTimeComplete`, and if set to false, clients should request the next segment after the last HTTP chunk of the previous Segment is received.

7. The Clients should implement an appropriate joining algorithm to support fast enough random access and at the same time meet the latency constraints. The following aspects should be considered:
    a. If Random Access within Segments is signaled (see below item 8), then client should use such random access points in order to join within the joining time as well as to keep the client in the target latency.
    b. If no such Random Access points are present, or if they are too far spread apart to to meet the latency requirements, the client is recommended, depending on the wall-clock time to
        i. Either download the latest available Segment from the beginning, but then do seek or an accelerated decoding to reach the time. Note that this may need a client that supports faster than real-time decoding.
        ii. Or to wait for the next segment to appear if this is expected to happen within the random access bounds and then request and start decoding the new Segment. If the latency is too short and the client supports decelerated decoding, it may apply this to reach the target latency.
    c. The guidelines on joinin a low latency stream in DVB DASH, clause 10.20.7, should be considered.
    d. Suitable combination of the above.

8. Clients should implement mechanisms to use Resynchronization Point information in operation.
    a. If a **`Resync`** element is present, clients should support finding Resynchronization Points in Segments using the functionality described in ISO/IEC 23009-1:2020/Amd.1 [X8], Annex A.12.3.
    b. If the **`Resync`** element signals random access points within Segments, clients should use this information in order to join a Representation such that both, access times and latency requirements are supported. This is documented in ISO/IEC 23009-1:2020/Amd.1 [X8], Annex A.12.2.
    c. If one or several Representations are available for which the client can downswitch or resynchronize after losses or buffer underruns, the DASH client should implement means for early Resynchronization in the Segment.

9. Clients should implement an appropriate bandwidth estimation when requesting Segments that are not yet complete at the time of request. Clients should use such an algorithm when requesting Segments of Low Latency Chunked Adaptation Sets. For more details, refer to clause 9.X.7.

10. Clients should implement appropriate failover mechanisms to resume to the target latency. For more details, refer to clause 9.X.7.

11. Clients should implement the Buffer management and drift avoidance functions as defined in clause 10.20.8 of the DVB DASH specification.

12. Clients should follow the Content Protection relevant recommendations and guidelines as provided in clause X.

9.X.6 Guidelines for Low-Latency Service Offering (Informative)

9.X.6.1 Introduction

This clause provides further guidelines and considerations for a Low-Latency Service Offering beyond the requirements and recommendations in clause 9.X.4. It is not meant to provide a normative implementation but provides a reference implementation as well as a set of guidelines on how to operate a headend for low-latency distribution.

Note: This clause is a starting point, more information is expected to be added over time.

9.X.6.2 ABR Encoder, Encryption and MP4/CMAF Packaging

9.X.6.2.1 General

This clause provides best practices on how to encode and encrypt content for low-latency DASH streaming. In quite many circumstances, the ABR encoder is provided information by a contribution link that itself has encoded material and metadata included. Despite such contribution encoders and links are not a matter of this document, the following is recommended for contribution encoders and links:

1. They should avoid adding unnecessary latency
2. Ad insertion markers and other messages requiring changes in the encoding configuration should be provided in advance to the actual change.
3. The encoding should avoid providing virtual segmentation, this should be the duty of the ABR encoder. If provided and important, this virtual segmentation should be consistent in terms of segment durations.
4. Regular encoding patterns such as IDR frames should be signaled and be consistent.

For the ABR encoder generating CMAF chunks and CMAF fragments, the following is recommended:

1. The ABR encoder should be aware of the MP4/CMAF logic for chunk generation.
2. If there is a desire for running specific latencies and configuration, an ABR encoder encoder for low latency streaming should permit to configure the following parameters:
    - Maximum and nominal fragment duration: i.e., the nominal interval of Switch Points. Note that this duration may be chosen from a set of of selected parameters and may depend on, for example the frame rate of content.
    - Maximum and nominal chunk duration: i.e., the nominal maximum duration of the chunk. Note that this duration may be chosen from a set of of selected parameters and may depend on for example the frame rate of content.
    - The maximum bitrate/size of a Fragment for each CMAF Track.
    - The maximum bitrate/size of a Chunk for each CMAF Track.
    - The maximum variation of fragment durations
        i. Signalling of constant segment duration using `@duration`, permitting a variation of `+/- max_variation` in percent value of the segment duration.
        ii. For each media segment in each Representation the MPD start time of the segment should approximately be EPT - PTO.
        iii. Specifically, the MPD start time of the segment is preferably be in the range of EPT - PTO – max_variation/100*DUR and EPT - PTO + max_variation*DUR, otherwise the DASH packager needs to add a new Period.
3. The ABR encoder should be able to produce CMAF Tracks within a Switching Set for which chunks start
    - with a SAP type 1, 2 or 3, i.e. permitting downswitching, resynchronization and random access.
    - with a SAP type 0, i.e. no restrictions.

9.X.6.2.2 Encoding and CMAF Chunk Duration Recommendations

9.X.6.2.2.1 General

CMAF chunks is a way to reduce streaming latency without decreasing the IDR frame frequency and the DASH segment sizes. There have been demonstrations using chunk sizes of one video frame, but in the general case if for example typical efficient encoding configurations with B-frames are used, then creating chunks with duration of 1 sample per chunk is not desirable. This does not only apply to video, but also for audio and possibly subtitles, it needs to be taken into consideration. For practical use, it is important to find an appropriate chunk duration for all media types. This following text is a first set of recommendations on this matter.

One reason for using chunks instead of shorter Segments is that Segments must start with a SAP type 1 or 2, i.e., for video a closed GOP. This decreases the coding efficiency of video. CMAF Chunks do not have this restriction since they are generated for neither bitrate switching nor randomly accessing the Representation.

Please be aware of the following:

1. CMAF chunks are generated such that the range of *decode* times contained in any CMAF chunk of the CMAF Track does not overlap with the range of *decode* times in any other CMAF chunk of the same CMAF Track.
2. Furthermore, CMAF chunks should be generated such that the range of presentation times contained in any CMAF chunk of the CMAF Track does not overlap with the range of presentation times in any other CMAF chunk of the same CMAF Track.

The content authoring, in particular the CMAF Chunk duration and the CMAF Fragment duration as well as the Target latency, impacts all of the three, (i) achievable latency, (ii) switching granularity and (iii) startup delay. A content author should carefully offer content such that the client can meet the desired parameters provided in the service description.

> Note: In a future version more considerations on how RAP frequency, chunk size and segment size impact the switching, end-to-end latency as well as the start-up latency. Those desired values determine proper chunk size as well as proper fragment duration/RAP distance).

9.X.6.2.2.2 Video

Applying the above rules, to make each chunk displayable, without waiting for more data, it is important that all B-frames which should be displayed before the P-frame are included in the chunk.

As an example, hierarchical B-frames with the display order

I0 B1 B2 B3 P4 | B5 B6 B7 P8 | B9 B10 B11 P12 | B13 B14 B15 P16 |

have a send (decode) order (order in mdat) like

I0 P4 B1 B2 B3 | P8 B6 B5 B7 | P12 B10 B9 B10 | P16 B14 B13 B15 |

Thus, here the segment should be broken after a multiple of 4-5 frames (160-200ms for 25Hz video). The general recommendation is therefore to put chunk boundaries so that all display times (decode time + composition_time_offset) are before the display-time of the earliest display time of the next chunk.

Depending on the encoder, this is either a fixed or variable duration.

This recommendation results that during the process of packaging the encoded video when packed into CMAF chunks does not add further delay beyond that arising from frame re-ordering in the encoder, helping to make chunks available at the earliest moment that they could be.

Furthermore, it is recommended to generate at least one Representation for which more frequent Random Access points are added, for example for each chunk or every second at least. Such Representations, if properly announced, may be used by clients for multiple purposes:

1. Fast random access while maintaining low latency
2. Quick Resynchronization after buffer underruns
3. In-Segment downswitching in case buffer draining is observed

9.X.6.2.2.3 Audio

For audio, one could make similar chunks as for video, but there is an issue that the audio bitrate is relatively low. For example, the size of an 200ms audio chunk of 64kbps is 1.6kB. That may possibly be too small to propagate through network or receiver buffers. For this reason, it may make sense to have longer chunks (e.g. 0.5s for audio) or even not applying chunking for audio but run at shorter Segment duration.

9.X.6.2.2.4 Subtitles

Subtitles is also low bitrate, but has different characteristics compared to audio. In particular, for IMSC-1 there is typically only one sample in each segment. However, that sample is a TTML XML document with relatively big boilerplate resulting in a size of roughly 2kB even though the actual information is just one short sentence or no text at all. For 2s segments, the bitrate will then be in the order of 10kbps. However, a chunk also needs to have one sample, so a chunk will have approximately the same size as an ordinary segment. If we would split a 2s subtitle segment into 10 chunks the bitrate would therefore increase from 10kbs to roughly 100kbps. Furthermore, it would increase the XML parsing in the client by a factor of 10. Since both these effects are undesirable, it is suggested that subtitles are not being chunked but delivered as separate segments every 1s or similar.

9.X.6.2.3 Producer Reference Time

The Producer Reference Timesupplies times corresponding to the production of associated media. This information permits among others to (i) provide media clients with information to enable consumption and production to proceed at equivalent rates, thus avoiding possible buffer overflow or underflow, and (ii) enable measuring and potentially controlling the latency between the production of the media time and the playout.

The Producer Reference Time (`'prft'`) as defined in ISO/IEC 14496-12.

NOTE: Please refer to the latest updates in N18416 [X6].

The information may be provided inband as part of the Segments in the (`'prft'`), in the MPD or both. In the context of the low-latency DASH service offerings, providing information in the MPD is strongly recommended, whereas providing inband information is left to the deployment.

If the CMAF Switching Set contains (`'prft'`) information and the `flags` is set to 0 or the `flags` 8 and 16 are set, then this information should be exposed to the MPD by adding a Producer Reference Time element **`ProducerReferenceTime`** as follows:

- `@id` is set to a unique value in the context of the Media Presentation
- `@inband` is set to `true`
- `@type` to `encoder` for flag set to 0 or to captured for both flag 8 and flag 16 being set.

Regardless, whether the inband information is present or not, it is recommended to provide information in the MPD for the `@wallclockTime` and the `@presentationTime`.

Assume that a value wall-clock `WC` is known that corresponds to a presentation time `PT`, either by the availability of a pair of for `ntp_timestamp` and `media_time` as contained in a (`'prft'`) as defined in 8.16.5 of ISO/IEC 14496-12 or by other means. Also, it is assumed that the value of the `@presentationTimeOffset` PTO is known. The MPD packager should act as follows:

- derive the wall-clock time WCA that corresponds to the PTO, namely `WCA = WC + (PT – PTO)`
- convert the `WCA` into the format of a **`UTCTiming`** element format present in the MPD
- add this **`UTCTiming`** element into the Producer Reference Time element **`ProducerReferenceTime`**
- add the value of PTO to the `@presentationTime`
- add the value of WCA to the `@wallclockTime`

Note that multiple producer reference time elements may be added.

9.X.6.2.4 Content Conditioning at Splice Points

No specific issues for low-latency content are known.

9.X.6.2.5 Robustness

For robustness, the guidelines in clause 4.8 apply. Also, please take into account the deployment scenarios in clause 4.11.

9.X.6.2.6 Encryption

For protected content, please take into account the considerations in clause 7.

9.X.6.2.7 Service description

Annex K of ISO/IEC 23009-1 defines the DASH Service Description. In the DASH model in ISO/IEC 23009-1, the DASH client has significant control over the algorithms and user perception for a DASH service. The DASH client may for example decide on the applied rate adaptation algorithm, the buffer strategy, the buffer duration and the resulting latency and channel access times. However, by leaving all decisions to the client, this may result in inconsistent behavior as different client implementations may for example choose different strategies and therefore, as an example, one may observe significantly different latencies for the same service on different clients.

Hence, the Annex K defines the following a service description reference model for the client as shown below. The semantics of the service description and the associated keys to describe the service are provided in K.3. This is defined as an abstract set of APIs that can be used by the application, regardless on how the application received this information. The usage of the service description information may be left to client implementations, but it may also be the case that application standards formulate stronger requirements on the client in order to fulfil such service parameters. Service Descriptions can also be scoped for specific clients, for example those implementing specific rules, or clients in specific environments.

Figure X.X shows an extended client model that includes the ability to provide explicit service description information to the DASH access client. The information may origin from the service provider and may be delivered by application defined signalling, may be generated in the application or may be delivered by the MPD as defined in Annex K.4 of ISO/IEC 23009-1.

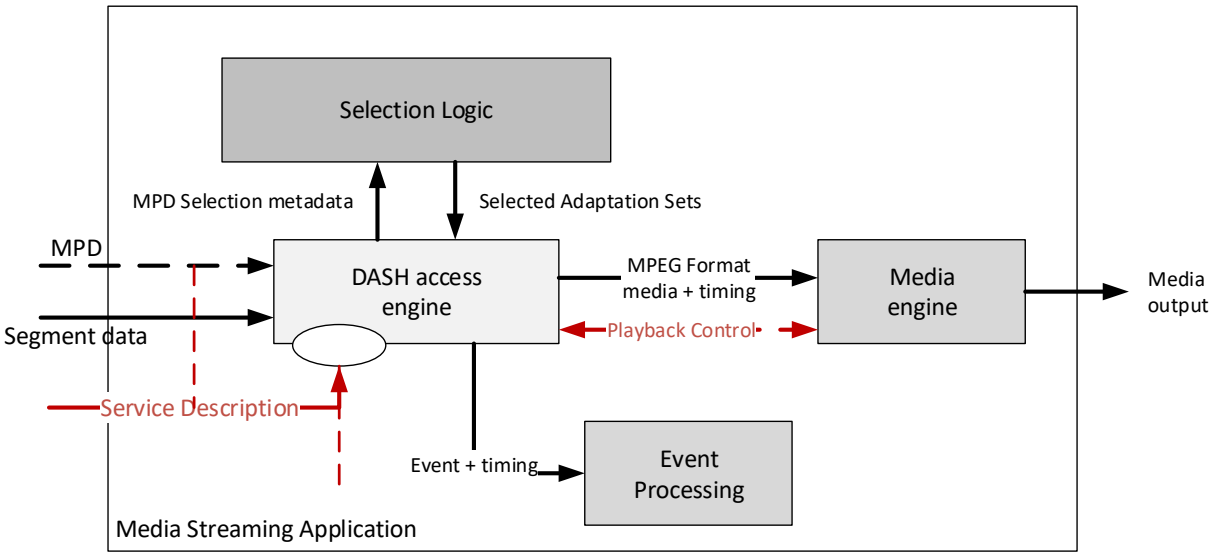

Figure X.X Extended Client Model with service description

In the context of LL-DASH applications, it is recommened that the MPD or the controlling application contains at least the following information with respect to latency targets for the client:

1. `TargetLatency`: The service provider's preferred presentation latency in milliseconds compared to the producer reference time. This information is strongly recommended.

2. `MaximumLatency`: The service provider's indication about the maximum presentation latency in milliseconds. Indicates a content provider's desire for the content not to be presented if the latency exceeds the maximum latency. If this applies for the service, then this information should be added
3. `MinimumLatency`: The service provider's indication about minimum presentation latency in milliseconds for example to avoid inconsistencies with second screen applications, overlays, etc. If this applies for the service, then this information should be added.
4. The latency should refer to a producer reference time in the MPD.

With respect to permitting DASH clients to do accelerated or decelerated playback, the **`PlaybackSpeed`** element may be present to provide a permission on the boundaries for this service. If present, the following nay be defined:

1. `max`: The maximum playback speed that the content provider indicates is appropriate for the purposes of automatically adjusting playback latency and buffer occupancy during normal playback.
2. `min`: The minimum playback speed that the content provider indicates is appropriate for the purposes of automatically adjusting playback latency and buffer occupancy during normal playback.

9.X.6.2.8 Resynchronization Points

Consistent insertion and signalling of Resynchronization Points is recommended, following ISO/IEC 23009-1:2020/Amd.1 [X8]. Signaling Resynchronization Points can be used by DASH clients for several purposes, namely

1) Fast random access while maintaining low latency
2) Quick Resynchronization after buffer underruns
3) In-Segment downswitching in case buffer draining is observed
4) Understanding the applied chunk size and duration and hence support for the rate adaptation.

The following is recommended:

1. Provide Resync signaling for each Low-Latency Chunked Adaptation Set by adding a **`Resync`** element with the following parameters (if known):
    - On Adaptation Set level
        - `@dT` providing the maximum and nominal duration of each chunk
        - `@dImax` providing the maximum size of a chunk. If unknown, parameter may be omitted.
        - `@dImin` providing the minimum size of a chunk. If unknown, parameter may be omitted.
        - `@type` is set to 0 to indicate that these are CMAF chunks without any promise for specific random-access capabilities beyond parsing.
2. Provide Resync signaling for each random-access Representation by adding a **`Resync`** element with the following parameters
    - On Representation level
        - `@dT` providing the maximum and nominal distance of two random access points.
        - `@dImax` providing the maximum size of the data in between the two random access points. If unknown, parameter may be omitted.
        - `@dImin` providing the minimum size of of the data in between the two random access points. If unknown, parameter may be omitted.
        - `@type` is set to 1, 2 or 3 to indicate that random access is possible.

9.X.6.2.9 Fast Switching Adaptation Sets

Fast switching Adaptation Set generation and signalling is for further study.

9.X.6.2.10 Leap Second Information

For details of settings and usage of Leap Second Information, please refer to DVB-DASH, clause 11.18.2.4.

9.X.6.2.11 MPD Generation for CMAF Media

It is recommended to apply the DASH Profile for CMAF content [X8] in case CMAF content is used.

9.X.6.3 Service Configuration Parameters for Low Latency

9.X.6.3.1 Introduction

This clause provides an informative overview on how service configuration parameters may be defined for an encoding and packaging workflow.

Selected Service Configuration Parameters are provided in clause 9.X.6.3.2. Clause 9.X.6.3.3 provides the mapping to DASH MPD parameters. Finally, clause 9.X.6.3.4 provides an overview on how FFMEG can be configured.

9.X.6.3.2 Selected Configuration Parameters

Table 1 provides an overview on proper configuration parameters that are useful or needed for an ABR encoder and a DASH Packager.

Table 1 Configuration Parameters needed for ABR Encoder and DASH Packager

Parameter	Symbol	Explanation
Low-Latency Presentation	`LL-FLAG`	Indicates that the media presentation is expected to conform to a DASH-IF Low latency presentation.
Target Latency	`TargetLatency`	See MPEG-DASH [X6], Annex K, Table K.1
Maximum Latency	`MaximumLatency`	See MPEG-DASH [X6], Annex K, Table K.1
Minimum Latency	`MinimumLatency`	See MPEG-DASH [X6], Annex K, Table K.1
Change Lead Time	`CLT`	the minimum time in media time that the DASH Packager gets a pre-warning for a change in the program offering, e.g. SCTE-35 messages lead time. This can be used for determining the minimum update period.
UTC Timing	`UTCTime`	All information about the UTC timing source.
Leap Second Information	`LeapSeconds`	Provides the information when the next leap second will happen and all relevant information.
Addressing Scheme	`AddScheme`	Indicated addressing scheme: `@duration` + $Number$, **`SegmentTimeline`** + $Number$, **`SegmentTimeline`** + $Time$,
Number of Adaptation/Switching Sets	`NoSS`	The number of Switching Sets that are generated by the encoder
Reference Buffer Duration	`RBD`	Defines a reference buffer duration for bandwidth information.
For each Adaptation/Switching Set `i=1, …, NoSS`		
Adaptation Set Type	`ASType`	Defines the Adaptation Set Type, namely a regular Adaptation Set not designed Low-Latency, a Low Latency Segment AS or a Low-Latency Chunked AS. Requirements for each of those Adaptation Set types are provided in clause 9.X.6.3.3.
MPD Validity Expiration	`MVE-Event`	Defines if the Adaptation Set carries the inband event messages for MPD validity expirations.

	Producer reference time	`PRRTF`	Indicates that the prft must be present in the media stream and that the ntp_time provides • `encoder` provides a reference when the media time was input to an encoder following the exact definition in subclause 8.16.5 of ISO/IEC 14496-12 for flags set to 0. Note: Low-Latency Adaptation Sets always provide a Producer Reference Time element in the MPD.
	CMAF Master Header	`CH*[i]`	The describing MP4/CMAF Header, i.e. the one that provides a compatibility point for all MP4/CMAF Tracks in the MP4/CMAF Switching Set. The MP4/CMAF header allows to extract the following parameters at the minimum: • Timescale • Media Type/Handler • Codecs Parameter (Sample entry) • Encryption Includes the below information
	Reference Buffer		Provides buffer information for the CMAF master header and the highest bitrate.
	Reference Bandwidth	`RBW[i]`	Consider a hypothetical constant bitrate channel of bandwidth with the value of this attribute in bits per second (bps). Then, if the MP4/CMAF Track is continuously delivered at this bitrate, starting at any Fragment Boundary a client can be assured of having enough data for continuous playout providing playout begins after `RBD[i]` * `RBW[i]` bits have been received (i.e. at time `RBI[i]` after the first bit is received).
	Nominal CMAF Fragment/Segment Duration	`SD[i]`	The nominal MP4/CMAF Fragment duration in the Adaptation/Switching Set (note that this is identical in terms of presentation time and in terms of decode time).
	Nominal MP4/CMAF chunk duration	`CD[i]`	The target MP4/CMAF chunk duration in decode time. This is in scale of the timescale.
	Number of CMAF Tracks/Representations in AS	`NT[i]`	Provides the number of CMAF Tracks provided in the Switching.
	Bitrate ladder percentage or bitrates of each Representation.	`BLP[i]`	Provides the bitrate reduction in percentage of each of the provided tracks compared to the master header. Or it provides the remaining bitrates. It is up to the packager to optimize encoding, for example using temporal subsampling.

9.X.6.3.3 Additional Configurations for Low-Latency Adaptation Set Types

For Low-Latency Segment Adaptation Sets, the following applies beyond Table X:

1. The chunk duration CD[i] is identical to the Segment Duration SD[i] and hence not provided.
2. The Segment Duration SD[i] is at most 30% of the Target Latency.

For Low-Latency Chunked Adaptation Sets with multiple Representations, the following applies beyond Table X:

1. If the Segment Duration is larger than 50% of the TargetLatency, then a Representation is added to the Adaptation Set with the following parameters
    - The bitrate is identical to the bitrate of the lowest bitrate Representation
    - If the nominal chunk duration is 500ms or more, then
        - Every Chunk is started with a SAP type 1

      Otherwise
        - A SAP type 1 is added to chunks such that the distance of the SAP types is between 500ms and 1s
2. The Chunks with a SAP type 1 create a Resync Marker Point as defined in MPEG DASH Amd.1.

9.X.6.3.4 Recommended Default Settings

Table 2 provides an overview on proper configuration parameters that are useful or needed for an ABR encoder and a DASH Packager.

Table 2 Proposed Default Parameters

Parameter	**Symbol**	**Proposed default Setting**
Low-Latency Presentation	`LL-FLAG`	Yes
Target Latency	`TargetLatency`	3.5 seconds
Maximum Latency	`MaximumLatency`	10 seconds
Minimum Latency	`MinimumLatency`	2 seconds
Change Lead Time	`CLT`	0
Reference Buffer Duration	`RBD`	1 second
UTC Timing	`UTCTime`	Akamai timing server with ms precision (<URL>)
Leap Second Information	`LeapSeconds`	Not present
Addressing Scheme	`AddScheme`	**`SegmentTimeLine`** + `$Number$`
Number of Adaptation/Switching Sets	`NoSS`	2
For each video Adaptation/Switching Set		
Adaptation Set Type	`ASType`	LL Chunked AS
MPD Validity Expiration	`MVE-Event`	Absent
Producer reference time	`PRRTF`	Absent
CMAF Master Header	`CH*[i]`	avc1
Reference Buffer		
Reference Buffer Duration	`RBD[i]`	6 seconds

		Reference Bandwidth	RBW[i]	**2 Mbit/s**
	Nominal CMAF Fragment/Segment Duration		SD[i]	6 seconds
	Nominal MP4/CMAF chunk duration		CD[i]	500ms
	Number of CMAF Tracks/Representations in AS		NT[i]	5
	Bitrate ladder percentage		BLP[i]	30%
For each audio Adaptation/Switching Set				
	Adaptation Set Type		ASType	LL Segment AS
	MPD Validity Expiration		MVE-Event	yes
	Producer reference time		PRRTF	yes
	CMAF Master Header		CH*[i]	aac
	Reference Buffer			
		Reference Buffer Duration	RBD[i]	1 second
		Reference Bandwidth	RBW[i]	**128 kbit/s**
	Nominal CMAF Fragment/Segment Duration		SD[i]	1 second
	Nominal MP4/CMAF chunk duration		CD[i]	n/a
	Number of CMAF Tracks/Representations in AS		NT[i]	1
	Bitrate ladder percentage		BLP[i]	n/a

9.X.6.3.5 DASH Mapping of Parameters

In the following a mapping of the information of Low-Latency parameters to the MPD is provided in Table 3.

Table 3 MPD Mapping of Low-Latency Specific Service Parameters

MPD Information	**Status**	**Proposed Value**
MPD@type	mandatory, set to "dynamic"	the type of the Media Presentation is dynamic, i.e. Segments get available over time.
MPD@availabilityStartTime	mandatory	the start time is the anchor for the MPD in wall-clock time. The value is denoted as *AST*.
MPD@minimumUpdatePeriod	mandatory	this field is mandatory for a continuous live service in order to indicate that the MPD may change on the server. This value needs to be set low enough such that with the CLT you can announce MPD

			updates fast enough. If unsure on the value, set it to 0.
MPD@minBufferTime		Mandatory	Set to RDB
ServiceDescription.Latency		Mandatory	Latency Parameters for Service Description
	@target	Mandatory	TargetLatency
	@min	Optional	MinLatency
	@max	Optional	MaxLatency
ServiceDescription.PlaybackSpeed		Optional	Playback Speed Parameters for Service Description
	@max	Optional	Maximum Playback Speed
	@min	Optional	Minimum Playback Speed
LeapSecondInformation		Recommended	details see clause 9.X.6.2.10
Period@start		mandatory	the start time of the Period relative to the MPD availability start time. The value is denoted as *PS*.
AdaptationSet			Low-Latency Chunked Adaptation using DASH Profile for CMAF. See clause 9.X.4.5 for details.
	ProducerReferenceTime	Mandatory	details see clauseclauses 9.X.4.3 and 9.X.6.2.3
	SegmentTemplate@media	Mandatory	the template for the Media Segment
	SegmentTemplate@startNumber	Optional optional default	the number of the first segment in the Period. The value is denoted as *SSN*.
	SegmentTemplate@duration		the duration of each Segment in units of a time. The value divided by the value of @timescale is denoted.
	Resync	Recommended	Details see clause 9.X.6.2.7
	SegmentBase@availabilityTimeOffset	Mandatory	Recommended to set to SD[i] – CD[i], but operation needs to ensure that chunks are made available in time.
	SegmentBase@availabilityTimeComplete	Mandatory	Set to TRUE.
	Representation@bandwidth	Mandatory	Set to RBW[i]
	Representation@qualityRanking	recommended	Set according to quality from 1 (best) to N (worst) and taking into account the service continuity Representation with additional SAP types 1.

AdaptationSet			Low-Latency Segment Adaptation using DASH Profile for CMAF. See clause 9.X.4.4 for details.
	ProducerReferenceTime	Mandatory	details see clauseclauses 9.X.4.3 and 9.X.6.2.3
	SegmentTemplate@media	Mandatory	the template for the Media Segment
	SegmentTemplate@startNumber	Optional	the number of the first segment in the Period. The value is denoted as *SSN*.
	SegmentTemplate@duration		the duration of each Segment in units of a time. The value divided by the value of @timescale is denoted.
	Representation@bandwidth	Mandatory	Set to RBW[i]

9.X.6.3.6 Example: FFMPEG Configuration

Note: This information is work in progress. DASH-IF together with DVB has commissioned work on this to update FFMPEG to support low-latency. The below mapping will be updated to reflect the latest status on how to generate LL-DASH with FFMPEG.

Producer reference time is present following the DVB-DASH and DASH-IF IOP specifications	*Present, not parsed by the validator*
Audio segments and video segments duration can be set with different values	*yes*
Chunked CMAF: the number of video frames or audio samples per CMAF chunk is freely configurable	The effect of the additional ffmpeg cli options “frag_duration” and “frag_type” need to be verified: { "frag_duration", "fragment duration (in seconds, fractional value can be set)", OFFSET(frag_duration), AV_OPT_TYPE_DURATION, { .i64 = 0 }, 0, INT_MAX, E } { "frag_type", "set type of interval for fragments", OFFSET(frag_type), AV_OPT_TYPE_INT, {.i64 = FRAG_TYPE_NONE }, 0, FRAG_TYPE_NB - 1, E, "frag_type"} { "none", "one fragment per segment", 0, AV_OPT_TYPE_CONST, {.i64 = FRAG_TYPE_NONE }, 0, UINT_MAX, E, "frag_type"} { "every_frame", "fragment at every frame", 0, AV_OPT_TYPE_CONST, {.i64 = FRAG_TYPE_EVERY_FRAME }, 0, UINT_MAX, E, "frag_type"} { "duration", "fragment at specific time intervals", 0, AV_OPT_TYPE_CONST, {.i64 = FRAG_TYPE_DURATION }, 0, UINT_MAX, E, "frag_type"} Need to check the default values for low latency mode
Chunked CMAF: the CMAF chunking can be automatically set to follow the frame reordering distance (P-frame distance)	The effect of the additional ffmpeg cli option frag_type=pframes needs to be verified: { "pframes", "fragment at keyframes and following P-Frame reordering (Video only, experimental)", 0, AV_OPT_TYPE_CONST, {.i64 = FRAG_TYPE_PFRAMES }, 0, UINT_MAX, E, "frag_type"} Confirmed

9.X.6.4 MPD Generator and Packager Operation for Low-Latency Content

9.X.6.4.1 General

This clause introduces a reference DASH packager for low-latency that also adds a period boundary at indicated times. The content at such program changes is expected to be properly formatted. The implementation only provides an example and different way may be used to accomplish low-latency.

The basic operation of the DASH packager is as follows:

- The DASH packager creates an initial MPD based on the configuration information in clause 9.X.6.3.
- DASH packager acts as a slave to the ABR encoder and incoming data formats on IF-1.
- The DASH packager formats and generates MPDs dynamically as well ingests the Segments into the CDN.
    - The DASH packager ensures the Segment availability times in the MPD are correct at the origin.
    - The DASH packager manages upstream triggers and generates the MPD and/or inserts events into the MPD and/or into the Segments.
- The DASH packager manages upstream errors, for example missing or corrupt data, etc.

The DASH packager communicates with the origin such that it can transfer Segments into the CDN, such that

1) The Segment URL is accessible even if only parts of the Segments are delivered
2) The Segment can be delivered in a progressive manner or chunked manner whereby the packager controls the chunk size. An example is HTTP Transfer Encoding, but also other ingest protocols may fulfil this property.

This clause provides more details for different DASH modes.

9.X.6.4.2 Backward-Compatible Simple Live

The clause provides a reference implementation for a DASH packager using the following DASH features and configuration:

- the simple live profile (with `$Number$` and `@duration` for duration signalling is used)
- the DASH client can either use MPD updates following `@minimumUpdatePeriod` or may rely on MPD validity expirations sent inband.
- multi-period offerings are supported, either for ad insertion opportunities, program changes or for operational purposes

The following service offering is assumed:

1. Each CMAF fragment generates one DASH segment
2. Each CMAF chunk is offered as an HTTP Chunk. Note that this configuration should not be an assumption by the client implementation.

The following two processes are documented:

- generation of the initial MPD
- dynamic operation of the packager including MPD processing and Segment offering

Assuming that the DASH packager receives configuration parameters and MP4/CMAF or MP4/CMAF equivalent data as as introduced in clause 9.X.3.2. Then the initial MPD parameters may be generated as shown in Table 3.

For each MP4/CMAF Track in a Switching Set, the track is mapped to a Representation in an Adaptation Set. Assuming that the MP4/CMAF track is a MP4/CMAF Track with the properties documented above and this MP4/CMAF Track is received by the DASH packager in sequences of MP4/CMAF Headers and MP4/CMAF Chunks.

- The `@minimumUpdatePeriod` is set to a value that is equal or smaller than the change lead time `CLT` provided by the segment stream.
- With incoming MP4/CMAF Headers and MP4/CMAF Chunks, the DASH packager operates as follows
    - CMAF Header: A new Period is generated and the Adaptation Set values are set as follows:
        - The `@timescale` of the Adaptation Set is set to the timescale of the included media (look into the MP4/CMAF Header and extract the time scale from the track header)
        - The `@duration` attribute is set such that the nominal duration `nominalFragmentDuration` is documented in the MPD for this Adaptation Set.
        - `$Number$` is used of segment templating.
        - Other Adaptation Set relevant parameters are extracted from the MP4/CMAF Header for the Adaptation Set and for each Representation
    - For Low Latency Segment Adaptation Sets as well for each initial MP4/CMAF chunk of a Low Latency Chunked Adaptation Set:
        - if it is not the first segment in the period, determine the duration of the previous segment (summing the truns of the previous chunks)
        - If a prft if present, the media time and the corresponding wall-clock time are stored in the packager such that at a new period, this information can be added to the manifest
        - a new segment is generated in the MPD by the DASH packager and the DASH packager checks the validity of the MPD offering. If still valid, no changes to MPD are done. Only if changes are done that are no longer valid, a new MPD is written. Specifically,
            - The MPD start time of the new segment must be in the range of *EPT* - *PTO* - 0.5***DUR* and *EPT* - *PTO* + 0.5***DUR* with *DUR* the value of `@duration`.
            - If this is not fulfilled a new Period is written that includes the following:
                - The **`Period`**`@start` is set such that the MPD start time is correct.
                - The `@presentationTimeOffset` is set to the EPT of the first segment
                - The **`ProducerReferenceTime`** element is added following the instructions from clause 4.X.4.3.2.
                - The `@startNumber` is set to the first segment in the new Period.
                - The Adaptation Sets are continued by providing Period continuity signalling with each Adaptation Set.
                - The `@availabilityTimeOffset` is set.
        - The initial chunk is offered as a resource that is referenced by the URL announced in the MPD and sent as an HTTP chunk to the origin
    - for each non-initial MP4/CMAF chunk of a Low Latency Chunked Adaptation Set
        - Event Message Processing
            - If an emsg is present, the emsg is stored and added to the next segment in a consistent manner. The emsg may be kept in the chunk if it is targeted for the client.
        - The non-initial chunk is appended to the segment that is offered by the corresponding URL announced in the MPD and sent as an HTTP chunk to the origin
- Generally, it is assumed that the contribution signal and the encoder are always available. However, when an encoder fails for one or more specific Representations to generate the next chunk, then the DASH content generator may apply appropriate means as documented in clause X.X.

- when a program change is announced, generates a new MPD as follows:
    - The `@minimumUpdatePeriod` is set to `0`.
- When the program change occurs
    - Write a new MPD with all the parameters
    - Reset the `@minimumUpdatePeriod` is set to a value that is equal or smaller than the change lead time provided

9.X.6.4.3 Backward-Compatible Main Live

Note: This extension using the SegmentTimeline is provided in a future version. ➔ Addressed in MPEG 4th edition.

According to the 4th edition of DASH, the following applies.

The `@d` attribute shall represent the accurate presentation duration of the Segment except for the following case:

- The `@availabilityTimeOffset` is present for the Representation, and
- the `@availabilityTimeComplete` is set to `FALSE` for the Representation, and
- the **S** element is the one with the largest **S**`@t` value in the MPD, and
- The non-adjusted segment availability time of the Segment is greater than or equal to the publish time of the MPD;

This means that for such a Segment, the `@d` attribute may be updated with a new version of the MPD, or a new S element may be added to reflect a different duration of the Segment than the nominal one. A DASH Client should not rely on nominal segment durations for the derivation of Segment addresses in the case the `$Time$` identifier is present, but instead the duration of the Segment may be derived by parsing the Segment and determining the value or by updating the MPD after the non-adjusted segment availability time.

9.X.6.4.4 Broadcast TV Profile

Note: This extension using the Broadcast TV profile may be provided in a future version.

9.X.7 Client Implementation Guidelines (Informative)

9.X.7.1 Low-Latency DASH Client

9.X.7.1.1 General

The following issues should be considered for a low-latency client implementation.

1. It needs to be configurable on using the low-latency mode and what configuration to use, e.g. how aggressive it is in the latency.
2. Updates and considerations in the ABR logic and bandwidth estimations
3. Considerations start-up operations to maintain latency and start-up delays (joining needs to be done carefully, better to wait for the next RAP than playing out old stuff).
4. The client is configured by
    - Information in the MPD
    - External information, for example a controlling app ➔ needs to have an app
    - Client does its own magic to identify proper configuration
    - Combinations of the above
    - Obeying the service description

Note: input is welcome.

9.X.7.1.2 Bandwidth Estimation

If the channel bandwidth available for downloading media segments is lower than the required bandwidth of all the Representations being played, this will lead to reduced buffer occupancy at the player and the need for downward adaptation can be detected straightforwardly. Excess channel bandwidth is more difficult to observe, as explained in more detail in “Bandwidth prediction in low-latency chunked streaming” [i.7]. One possible approach could involve measuring over HTTP/1.1 “chunked transfer encoding” chunks or HTTP/2 frames, provided these are of sufficient size. If the HTTP chunk/frame structure is not visible (e.g. in a browser-based client using the Fetch API) or the chunks or frames are very short, data throughput might be measured over the course of individual CMAF chunks (by incrementally parsing the data). Another approach could involve probing occasionally to determine if additional capacity is available. Probes will be most reliable if they request data that is cached at the CDN, such as part of a recent media segment from a Representation being played. Players could choose the portion of such a segment to request based on an expected delivery duration that would be measurable but which would not significantly impact the player’s buffer. These different approaches are explained and compared in terms of performance in “Bandwidth prediction in low-latency chunked streaming” [i.7].

Note: input is welcome.

9.X.7.1.3 Backward-Compatible Simple Live

Note: Specific issues will be added during a future revision

9.X.7.1.4 Backward-Compatible Main Live

Note: Specific issues will be added during a future revision

9.X.7.1.5 Broadcast TV Profile

Note: Specific issues will be added during a future revision

9.X.7.2 Reference Playback of Low-Latency

The reference playback platform is expected to support the CTA WAVE Device Playback Specification requirements [X5], in particular playback of low-latency and chunked content.

9.X.8 Network Operations Guidelines (Informative)

The following issues should be considered when operating an low latency service on a CDN.

1. Origin Server (push or pull)
    - Transfer encoding starts from here
    - TLS encryption will be applied from here
2. CDN operation
    - Special operations as well
    - Need to keep transfer functions in place
    - And need to cache the object
    - Needs to run with https
    - New HTTP codes? Such as come back later
3. Access Network
    - Is preferably stable in throughput
4. Network protocols and configuration, CDN operation (new status codes, etc.), etc. This may include recommendations to use push-protocols (e.g. HTTP/2, WebRTC) to reduce latencies

Note: input is welcome.