Guidelines for Implementation: DASH-IF Interoperability Points

September 07, 2017

DASH Industry Forum

Version 4.1

Scope

The scope of the DASH-IF InterOperability Points (IOPs) defined in this document is to provide support for high-quality video distribution for over the top services using H.264/AVC and H.265/HEVC. Both live and on-demand services are supported. The specified features enable relevant use cases including on-demand, live services, ad insertion, trick modes, seek preview, content protection and subtitling. Extensions for multi-channel audio and next generation audio with different codecs as well as extensions to video different codecs and Ultra High Definition are defined.

Any identified bugs or missing features may be submitted through the DASH-IF issue tracker at https://gitreports.com/issue/Dash-Industry-Forum/DASH-IF-IOP.

Note that version 4.1 is published as an add on to v4.0, but with the next version it is expected that a either a multipart version or a restructured version will be generated, with a major editorial updates. The new version is expected to be available by the end of 2017.

Disclaimer

This is a document made available by DASH-IF. The technology embodied in this document may involve the use of intellectual property rights, including patents and patent applications owned or controlled by any of the authors or developers of this document. No patent license, either implied or express, is granted to you by this document. DASH-IF has made no search or investigation for such rights and DASH-IF disclaims any duty to do so. The rights and obligations which apply to DASH-IF documents, as such rights and obligations are set forth and defined in the DASH-IF Bylaws and IPR Policy including, but not limited to, patent and other intellectual property license rights and obligations. A copy of the DASH-IF Bylaws and IPR Policy can be obtained at http://dashif.org/.

The material contained herein is provided on an "AS IS" basis and to the maximum extent permitted by applicable law, this material is provided AS IS, and the authors and developers of this material and DASH-IF hereby disclaim all other warranties and conditions, either express, implied or statutory, including, but not limited to, any (if any) implied warranties, duties or conditions of merchantability, of fitness for a particular purpose, of accuracy or completeness of responses, of workmanlike effort, and of lack of negligence.

In addition, this document may include references to documents and/or technologies controlled by third parties. Those third party documents and technologies may be subject to third party rules and licensing terms. No intellectual property license, either implied or express, to any third party material is granted to you by this document or DASH-IF. DASH-IF makes no any warranty whatsoever for such third party material.

Note that technologies included in this document and for which no test and conformance material is provided, are only published as a candidate technologies, and may be removed if no test material is provided before releasing a new version of this guidelines document. For the availability of test material, please check http://www.dashif.org.

Contents

List of Figures

List of Tables

Acronyms, abbreviations and definitions

For acronyms, abbreviations and definitions refer to ISO/IEC 23009-1 [4]. Additional definitions may be provided in the context of individual sections.

In addition, the following abbreviations and acronyms are used in this document:

AAC	Advanced Audio Coding
AFD	Active Format Description
AST	Availability Start Time
AVC	Advanced Video Coding
BL	Base Layer
BMFF	Base Media File Format
CDN	Content Delivery Network
CEA	Consumer Electronics Association
CT	Composition Time
DECE	Digital Entertainment Content Ecosystem
DRM	Digital Rights Management
DSI	Decoder Specific Information
DT	Decode Time
DTV	Digital Television
DVB	Digital Video Broadcasting
DVS	Digital Video Subcommittee
ECL	Entitlement Control License
EDL	Encoding Decision List
EL	Enhancement Layer
EME	Encrypted Media Extension
EML	Entitlement Management License
EPT	Earliest Presentation Time
FCC	Federal Communications Commission
GOP	Group-of-Pictures
HD	High-Definition

HDR	High Dynamic Range
HDR10	DASH-IF HDR 10 bit
HDMI	High-Definition Multimedia Interface
HE-AAC	High Efficiency AAC
HEVC	High-Efficiency Video Coding
KID	common Key Identifier
IAB	International Advertising Bureau
IDR	Instantaneous Decoder Refresh
IOP	InterOperability Point
ISO	International Standards Organization
HDR	High Dynamic Range
HEVC	High Efficiency Video Coding
HFR	High Frame Rate
HTTP	HyperText Transport Protocol
MBT	Minimum Buffer Time
MHA	MPEG-H 3D Audio
MPEG	Moving Pictures Experts Group
MUP	Minimum Update Period
NAL	Network Abstraction Layer
OTT	Over-The-Top
PCM	Pulse Code Modulation
PIFF	Protected Interoperable File Format
PPS	Picture Parameter Set
PQ	Perceptual Quantization
PS	Parametric Stereo
PT	Presentation Time
PTO	Presentation Time Offset
PVR	Personal Video Recorder
RFC	Request for Comments
SAP	Stream Access Point
SAET	Segment Availability End Time
SAST	Segment Availability Start Time
SBR	Spectral Band Replication
SCTE	Society of Cable Telecommunications Engineers

SD	Standard Definition
SDR	Standard Dynamic Range
SEI	Supplemental Enhancement Information
SMPTE	Society of Motion Picture and Television Engineers
SPD	Suggested Presentation Delay
SPS	Sequence Parameter Set
TSB	Time Shift Buffer depth
TT	Timed Text
TTML	Timed Text Markup Language
UHD	Ultra-High Definitions
URL	Universal Resource Location
UTC	Universal Time Clock
UUID	Universally Unique Identifier
VAST	Video Ad Serving Template
VES	Video Elementary Stream
VP9	Video Project 9
VPS	Video Parameter Set
VUI	Video Usability Information
WCG	Wide Colour Gamut

References

Notes:

1) If appropriate, the references refer to specific versions of the specifications. However, implementers are encouraged to check later versions of the same specification, if available. Such versions may provide further clarifications and corrections. However, new features added in new versions of specifications are not added automatically.
2) Specifications not yet officially available are marked in *italics*.
3) Specifications considered informative only are marked in Arial

[1] DASH-IF DASH-264/AVC Interoperability Points, version 1.0, available at http://dashif.org/w/2013/06/DASH-AVC-264-base-v1.03.pdf

[2] DASH-IF DASH-264/AVC Interoperability Points, version 2.0, available at http://dashif.org/w/2013/08/DASH-AVC-264-v2.00-hd-mca.pdf

[3] ISO/IEC 23009-1:2012/Cor.1:2013 Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment formats.

Note: this document is superseded by reference [4], but maintained as the initial version of this document is provided in the above reference.

[4] ISO/IEC 23009-1:2014 Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment formats. Including:

ISO/IEC 23009-1:2014/Cor 1:2015

ISO/IEC 23009-1:2014/Cor 2:2015

ISO/IEC 23009-1:2014/Cor 3:2017

ISO/IEC 23009-1:2014/Amd 1:2015 High Profile and Availability Time Synchronization

ISO/IEC 23009-1:2014/Amd 2:2015 Spatial relationship description, generalized URL parameters and other extensions

ISO/IEC 23009-1:2014/Amd 3:2016 Authentication, MPD linking, Callback Event, Period Continuity and other Extensions.

ISO/IEC 23009-1:2014/Amd 4:2016 Segment Independent SAP Signalling (SISSI), MPD chaining, MPD reset and other extensions [Note: Expected to be published by end of 2017. The Study of DAM is available in the MPEG output document w16221.]

All the above is expected to be rolled into a third edition of ISO/IEC 23009-1 as:

ISO/IEC 23009-1:2017 Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment formats. [Note: Expected to be published by end of 2017. The draft third edition is available in the MPEG output document w1467.]

[5] ISO/IEC 23009-2:2014: Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 2: Conformance and Reference.

[6] ISO/IEC 23009-3:2014: Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 3: Implementation Guidelines.

[7] ISO/IEC 14496-12:2015 Information technology -- Coding of audio-visual objects -- Part 12: ISO base media file format.

[8] ITU-T Recommendation H.264 (01/2012): "Advanced video coding for generic audiovisual services" | ISO/IEC 14496-10:2010: "Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding".

[9] ISO/IEC 14496-15:2014/Cor 1:2015: Information technology -- Coding of audio-visual objects -- Part 15: Carriage of network abstraction layer (NAL) unit structured video in ISO base media file format.

[10] IETF RFC 6381, The 'Codecs' and 'Profiles' Parameters for "Bucket" Media Types, August 2011.

[11] ISO/IEC 14496-3:2009 - Information technology -- Coding of audio-visual objects -- Part 3: Audio with Corrigendum 1:2009, Corrigendum 2:2011, Corrigendum 3:2012, Amendment 1:2009, Amendment 2:2010, Amendment 3:2012, and Amendment 4:2014.

[12] ISO/IEC 14496-14:2003/Amd 1:2010 Information technology -- Coding of audio-visual objects -- Part 14: The MP4 File Format

[13] 3GPP (2005-01-04). "ETSI TS 126 401 V6.1.0 (2004-12) - Universal Mobile Telecommunications System (UMTS); General audio codec audio processing functions; Enhanced aacPlus general audio codec; General description (3GPP TS 26.401 version 6.1.0 Release 6)"

[14] ANSI/CEA-708-E: Digital Television (DTV) Closed Captioning, August 2013

[15] 3GPP TS 26.245: "Transparent end-to-end Packet switched Streaming Service (PSS); Timed text format"

[16] W3C Timed Text Markup Language 1 (TTML1) (Second Edition) 24 September 2013.

[17] SMPTE ST 2052-1:2013 "Timed Text Format (SMPTE-TT)", https://www.smpte.org/standards

[18] W3C WebVTT - The Web Video Text Tracks,— http://dev.w3.org/html5/webvtt/

[19] ITU-T Recommendation H.265 (04/2015): "Advanced video coding for generic audiovisual services" | ISO/IEC 23008-2:2015/Amd 1:2015: " High Efficiency Coding and Media Delivery in Heterogeneous Environments – Part 2: High Efficiency Video Coding", downloadable here: http://www.itu.int/rec/T-REC-H.265

[20] EBU Tech 3350, "EBU-TT, Part 1, Subtitling format definition", July 2012, http://tech.ebu.ch/docs/tech/tech3350.pdf?vers=1.0

[21] IETF RFC 7230, Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing, June 2014.

[22] IETF RFC 7231, Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content, June 2014.

[23] IETF RFC 7232, Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests, June 2014.

[24] IETF RFC 7233, Hypertext Transfer Protocol (HTTP/1.1): Range Requests, June 2014.

[25] IETF RFC 7234, Hypertext Transfer Protocol (HTTP/1.1): Caching, June 2014.

[26] IETF RFC 7235, Hypertext Transfer Protocol (HTTP/1.1): Authentication, June 2014.

[27] SMPTE RP 2052-10-2013: Conversion from CEA-608 Data to SMPTE-TT https://www.smpte.org/standards

[28] SMPTE RP 2052-11-2013: Conversion from CEA 708 to SMPTE-TT https://www.smpte.org/standards

[29] ISO/IEC 14496-30:2014, "Timed Text and Other Visual Overlays in ISO Base Media File Format". Including:

ISO/IEC 14496-30:2014, Cor 1:2015

ISO/IEC 14496-30:2014, Cor 2:2016

[30] ISO/IEC 23001-7:2016: "Information technology -- MPEG systems technologies -- Part 7: Common encryption in ISO base media file format files".

[31] DASH Industry Forum, Test Cases and Test Vectors: http://testassets.dashif.org/.

[32] DASH Industry Forum, "Guidelines for Implementation: DASH-AVC/264 Conformance Software", http://dashif.org/conformance.html.

[33] DASH Identifiers Repository, available here: http://dashif.org/identifiers

[34] DTS 9302J81100, "Implementation of DTS Audio in Media Files Based on ISO/IEC 14496", http://www.dts.com/professionals/resources/resource-center.aspx

[35] ETSI TS 102 366 v1.2.1, Digital Audio Compression (AC-3, Enhanced AC-3) Standard (2008-08)

[36] MLP (Dolby TrueHD) streams within the ISO Base Media File Format, version 1.0, September 2009.

[37] ETSI TS 102 114 v1.3.1 (2011-08), "DTS Coherent Acoustics; Core and Extensions with Additional Profiles"

[38] ISO/IEC 23003-1:2007 - Information technology -- MPEG audio technologies -- Part 1: MPEG Surround

[39] DTS 9302K62400, "Implementation of DTS Audio in Dynamic Adaptive Streaming over HTTP (DASH)", http://www.dts.com/professionals/resources/resource-center.aspx

[40] IETF RFC5905, "Network Time Protocol Version 4: Protocol and Algorithms Specification," June 2010.

[41] IETF RFC 6265: "HTTP State Management Mechanism", April 2011.

[42] ETSI TS 103 285 v.1.1.1: "MPEG-DASH Profile for Transport of ISO BMFF Based DVB Services over IP Based Networks".

[43] ANSI/SCTE 128-1 2013: "AVC Video Constraints for Cable Television, Part 1 - Coding", available here: http://www.scte.org/documents/pdf/Standards/ANSI_SCTE%20128-1%202013.pdf

[44] IETF RFC 2119, "Key words for use in RFCs to Indicate Requirement Levels", April 1997.

[45] ISO: "ISO 639.2, Code for the Representation of Names of Languages — Part 2: alpha-3 code," as maintained by the ISO 639/Joint Advisory Committee (ISO 639/JAC), http://www.loc.gov/standards/iso639-2/iso639jac.html; JAC home page: http://www.loc.gov/standards/iso639-2/iso639jac.html; ISO 639.2 standard online: http://www.loc.gov/standards/iso639-2/langhome.html.

[46] CEA-608-E, Line 21 Data Service, March 2008.

[47] IETF RFC 5234, "Augmented BNF for Syntax Specifications: ABNF", January 2008.

[48] SMPTE ST 2086:2014, "Mastering Display Color Volume Metadata Supporting High Luminance And Wide Color Gamut Images"

[49] ISO/IEC 23001-8:2016, "Information technology -- MPEG systems technologies -- Part 8: Coding-independent code points", available here: http://standards.iso.org/ittf/PubliclyAvailableStandards/c069661_ISO_IEC_23001-8_2016.zip

[50] IETF RFC 7164, "RTP and Leap Seconds", March 2014.

[51] void

[52] IAB Video Multiple Ad Playlist (VMAP), available at http://www.iab.net/media/file/VMAPv1.0.pdf

[53] IAB Video Ad Serving Template (VAST), available at http://www.iab.net/media/file/VASTv3.0.pdf

[54] ANSI/SCTE 35 2015, Digital Program Insertion Cueing Message for Cable

[55] ANSI/SCTE 67 2014, Recommended Practice for SCTE 35 Digital Program Insertion Cueing Message for Cable

[56] ANSI/SCTE 214-1, MPEG DASH for IP-Based Cable Services, Part 1: MPD Constraints and Extensions

[57] ANSI/SCTE 214-3, MPEG DASH for IP-Based Cable Services, Part 3: DASH/FF Profile

[58] EIDR ID Format - EIDR: ID Format, v1.2, March 2014, available at http://eidr.org/documents/EIDR_ID_Format_v1.2.pdf

[59] Common Metadata, TR-META-CM, ver. 2.0, January 3, 2013, available at http://www.movielabs.com/md/md/v2.0/Common_Metadata_v2.0.pdf

[60] IETF RFC 4648, "The Base16, Base32, and Base64 Data Encodings", October 2006.

[61] W3C TTML Profiles for Internet Media Subtitles and Captions 1.0 (IMSC1), Editor's Draft 03 August 2015, available at: https://dvcs.w3.org/hg/ttml/raw-file/tip/ttml-ww-profiles/ttml-ww-profiles.html

[62] W3C TTML Profile Registry, available at: https://www.w3.org/wiki/TTML/CodecsRegistry

[63] ETSI TS 103 190-1 v1.2.1, "Digital Audio Compression (AC-4); Part 1: Channel based coding".

[64] ISO/IEC 23008-3:2015 - Information technology -- High efficiency coding and media delivery in heterogeneous environments -- Part 3: 3D audio with Amendment 1:2015 and Amendment 2:2015

[65] IETF RFC 2818, "HTTP Over TLS", May 2000.

[66] IETF RFC 4337, "MIME Type Registration for MPEG-4", March 2006.

[67] SMPTE: "Digital Object Identifier (DOI) Name and Entertainment ID Registry (EIDR) Identifier Representations," RP 2079-2013, Society of Motion Picture and Television Engineers, 2013.

[68] SMPTE: "Advertising Digital Identifier (Ad-ID®) Representations," RP 2092-1, Society of Motion Picture and Television Engineers, 2015.

[69] W3C Encrypted Media Extensions - https://www.w3.org/TR/encrypted-media/.

[70] void

[71] SMPTE ST 2084:2014, “Mastering Display Color Volume Metadata Supporting High Luminance and Wide Color Gamut Images”

[72] ISO/IEC 23001-8:2013, “Information technology -- MPEG systems technologies -- Part 8: Coding-independent code points”, available here: http://standards.iso.org/ittf/PubliclyAvailableStandards/c062088_ISO_IEC_23001-8_2013.zip

[73] Recommendation ITU-R BT.709-6 (06/2015): "Parameter values for the HDTV standards for production and international programme exchange".

[74] Recommendation ITU-R BT.2020-1 (06/2014): "Parameter values for ultra-high definition television systems for production and international programme exchange".

[75] ETSI TS 101 154 v2.2.1 (06/2015): "Specification for the use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream."

[76] ETSI TS 103 285 v1.1.1 (05/2015): "Digital Video Broadcasting (DVB); MPEG-DASH Profile for Transport of ISO BMFF Based DVB Services over IP Based Networks.”

[77] 3GPP TS 26.116 (03/2016): "Television (TV) over 3GPP services; Video Profiles.”

[78] DECE (05/2015): “Common File Format & Media Formats Specification”, http://uvcentral.com/sites/default/files/files/PublicSpecs/CFFMediaFormat-2_2.pdf

[79] Ultra HD Forum: Phase A Guidelines, version 1.1, July 2015

[80] Recommendation ITU-R BT.2100-1 (07/2016): "Image parameter values for high dynamic range television for use in production and international programme exchange".

[81] SMPTE ST 2086:2014, “Mastering Display Color Volume Metadata Supporting High Luminance And Wide Color Gamut Images”

[82] SMPTE ST 2094-1:2016, “Dynamic Metadata for Color Volume Transform – Core Components”

[83] SMPTE ST 2094-10:2016, “Dynamic Metadata for Color Volume Transform – Application #1”

[84] Recommendation ITU-R BT.1886: “Reference electro-optical transfer function for flat panel displays used in HDTV studio production”

[85] ETSI DGS/CCM-001 GS CCM 001 “Compound Content Management”

[86] VP9 Bitstream & Decoding Process Specification. https://storage.googleapis.com/downloads.webmproject.org/docs/vp9/vp9-bitstream-specification-v0.6-20160331-draft.pdf

[87] VP Codec ISO Media File Format Binding https://www.webmproject.org/vp9/mp4/

1. Introduction

This document defines DASH-IF's InterOperability Points (IOPs). The document includes IOPs for only this version of the document. For earlier versions, please refer to version 1 [1] and version 2 [2] of this document. DASH-IF recommends to deprecate the IOPs in previous versions and deploy using one of the IOPs and extensions in this document.

As a historical note, the scope of the initial DASH-AVC/264 IOP, issued with version 1 of this document [1] was the basic support high-quality video distribution over the top. Both live and on-demand services are supported.

In the second version of this document [2], HD video (up to 1080p) extensions and several multichannel audio extensions are defined.

In this third version of the DASH-IF IOP document, two new DASH-264/AVC IOPs are defined. Detailed refinements and improvements for DASH-IF live services and for ad insertion were added in these IOPs. One of these IOP is the superset of the simpler one. Additionally, two corresponding IOPs are defined to also support HEVC [19]. In both cases, AVC and HEVC, the more advanced IOP adds additional requirements on the DASH client to support segment parsing to achieve enhancement of live services. This structuring separates the Media Profiles from DASH features.

In the fourth version, beyond minor improvements, corrections and alignment with MPEG-DASH third edition, the key additions are extensions for next generation audio and UHD/HDR video.

This document defines the IOPs in Table 1 and Extensions in Table 2. The Implementation Guideline's version in which each IOP or Extension was added is also provided in the tables.

Note that all version 1 IOPs are also defined in version 2 and therefore referencing version [2] is sufficient.

Table 1 DASH-IF Interoperability Points

Interoperability Point	Identifier	Version	Reference
DASH-AVC/264	`http://dashif.org/guidelines/dash264`	1.0	[2], 6.3
DASH-AVC/264 SD	`http://dashif.org/guidelines/dash264#sd`	1.0	[2], 7.3
DASH-AVC/264 HD	`http://dashif.org/guidelines/dash264#hd`	2.0	[2], 8.3
DASH-AVC/264 main	`http://dashif.org/guidelines/dash264main`	3.0	8.2
DASH-AVC/264 high	`http://dashif.org/guidelines/dash264high`	3.0	8.3
DASH-IF IOP simple	`http://dashif.org/guidelines/dash-if-simple`	3.0	8.4
DASH-IF IOP main	`http://dashif.org/guidelines/dash-if-main`	3.0	8.5

Note that all extensions defined in version 2 of this document are carried over into version 3 without any modifications. In order to maintain a single document, referencing in Table 2 is restricted to this document.

Table 2 DASH-IF Interoperability Point Extensions

Extension	Identifier	Version	Section

DASH-IF multichannel audio extension with Enhanced AC-3	`http://dashif.org/guidelines/dashif#ec-3`	2.0	9.4.2.3
DASH-IF multichannel extension with Dolby TrueHD	`http://dashif.org/guidelines/dashif#mlpa`	2.0	9.4.2.3
DASH-IF multichannel audio extension with DTS Digital Surround	`http://dashif.org/guidelines/dashif#dtsc`	2.0	9.4.3.3
DASH-IF multichannel audio extension with DTS-HD High Resolution and DTS-HD Master Audio	`http://dashif.org/guidelines/dashif#dtsh`	2.0	9.4.3.3
DASH-IF multichannel audio extension with DTS Express	`http://dashif.org/guidelines/dashif#dtse`	2.0	9.4.3.3
DASH-IF multichannel extension with DTS-HD Lossless (no core)	`http://dashif.org/guidelines/dashif#dtsl`	2.0	9.4.3.3
DASH-IF multichannel audio extension with MPEG Surround	`http://dashif.org/guidelines/dashif#mps`	2.0	9.4.4.3
DASH-IF multichannel audio extension with HE-AACv2 level 4	`http://dashif.org/guidelines/dashif#heaac-mc51`	2.0	9.4.5.3
DASH-IF multichannel audio extension with HE-AACv2 level 6	`http://dashif.org/guidelines/dashif#heaac-mc71`	2.0	9.4.5.3
DASH-IF multichannel extension with AC-4	`http://dashif.org/guidelines/dashif#ac-4`	3.1	9.4.2.3
DASH-IF multichannel audio extension with MPEG-H 3D Audio	`http://dashif.org/guidelines/dashif#mha1` `http://dashif.org/guidelines/dashif#mha2`	3.2	9.4.6.3
DASH-IF UHD HEVC 4k	`http://dashif.org/guidelines/dash-if-uhd#4k`	4.0	10.2
DASH-IF HEVC HDR PQ10	`http://dashif.org/guidelines/dash-if-uhd#hdr-pq10`	4.0	10.3
DASH-IF UHD Dual-Stream (Dolby Vision)	`http://dashif.org/guidelines/dash-if-uhd#hdr-pq10`	4.1	10.4
DASH-IF VP9 HD	`http://dashif.org/guidelines/dashif#vp9`	4.1	11.3.1
DASH-IF VP9 UHD	`http://dashif.org/guidelines/dash-if-uhd#vp9`	4.1	11.3.2
DASH-IF VP9 HDR	`http://dashif.org/guidelines/dashif#vp9-hdr` `http://dashif.org/guidelines/dash-if-uhd#vp9-hdr`	4.1	11.3.3

Test cases and test vectors for DASH-AVC/264 IOPs are provided in [31]. The conformance and reference software for DASH-IF IOPs is provided in [32] (based on the MPEG conformance software). DASH Identifiers for different categories can be found online [33]. Note that technologies included in this document and for which no test and conformance material is provided, are only published as a candidate technologies, and may be removed if no test material is provided before releasing a new version of this guidelines document.

In addition to the Interoperability points in Table 1 and extensions in Table 2, this document also defines several other identifiers and other interoperability values for functional purposes as documented in Table 3.

Table 3 Identifiers and other interoperability values defined in this Document

Identifier	Semantics	Type	Section
http://dashif.org/identifiers/vast30	Defines an event for signalling events of VAST3.0	Event	5.6
http://dashif.org/guidelines/trickmode	Defines a trick mode Adaptation Set.	Functionality	3.2.9
http://dashif.org/guidelines/clearKey	Defines name space for the Laurl element in W3C	Namespace	7.6.2.4
e2719d58-a985-b3c9-781a-b030af78d30e	UUID for W3C Clear Key with DASH	Content Protection	7.6.2.4
http://dashif.org/guidelines/last-segment-number	Signaling last segment number	Functionality	4.4.3.6
http://dashif.org/guidelines/thumbnail_tile	Signalling the availability of the thumbnail tile adaptation set	Functionality	6.2.6

Note that all identifiers are collected at and http://dashif.org/identifiers/ and external identifiers are encouraged to be submitted for documentation there as well. Note also that DASH-IF typically tries to avoid defining identifiers. Identifiers in *italics* are subject to discussion with other organizations and may be deprecated in a later version.

2. Context and Conventions

2.1. Relation to MPEG-DASH and other DASH specifications

Dynamic Adaptive Streaming over HTTP (DASH) is initially defined in the first edition of ISO/IEC 23009-1 which was published in April 2012 and some corrections were done in 2013 [1]. In May 2014, ISO/IEC published the second version of ISO/IEC 23009-1 [4] that includes additional features and provide additional clarifications. The initial two versions of this document where based on the first edition of ISO/IEC 23009-1. This version is based on the second edition of ISO/IEC 23009-1, i.e. ISO/IEC 23009-1:2014 including Cor.3 and Amd.3 [4]. This means that also for all interoperability points that were initially defined in earlier versions of the document, also now the second edition serves as the reference. Backward-compatibility across different edition is handled by MPEG-DASH in ISO/IEC 23009-1 [4]. Note that this document also refers to technologies in draft corrigenda and draft amendments of MPEG. For this version, in particular Draft Amd.4 of ISO/IEC 23009-1:2014, and Draft Cor.3 of ISO/IEC 23009-1:2014 are of relevance.

This document was generated in close coordination with DVB-DASH [42]. The tools and features are aligned to the extent considered reasonable. To support implementers, this document attempts to highlight any differences and/or further restrictions or extensions when compared to DVB-DASH. However, as a disclaimer, this coverage is not considered complete.

2.2. Compatibility and Extensions to Earlier Versions

2.2.1. Summary of Version 3 Modifications

Version 3 of this document applies the following modifications compared to version 2 [2]:

- Reference to the second edition of ISO/IEC 23009-1 including amendment 1 and cor.1 [4], as well as well as Amendment 3 [4].
- Add an explicit statement in DASH-264/AVC to forbid time code wrap around
- Definition on the usage of key words in clause 2.3.
- Add more constraints on the usage of Trick Modes for improved interoperability in clause 3.2.9.
- Add more constraints on the Representations in one Adaptation Set in clause 3.2.10, especially for the case when the bitstream switching is true.
- Add additional details on the usage of HTTP in clause 3.4.
- Add H.265/HEVC as a codec and create IOPs for inclusion of this codec.
- Add CEA-608/708 closed captioning in SEI messages in clause 6.4.3.
- Detailed description of simple and main live operation, with the latter including segment parsing in clause 4.
- Detailed description of server-based and app-based ad insertion in clause 5
- General editorial updates and clarifications
- Updates and clarification to clause 7 on DRM and common encryption.
- Update to references
- Relaxation of the audio encoding requirements in clause 6.3.2.
- Add clarification on the usage of the minimum buffer time and bandwidth in clause 3.2.8.
- Add an informative clause on the timing model of DASH in clause 3.2.7.
- Relax the use of the '`lmsg`' brand for signaling the last segment in clause 3.6.
- Simplification of the codecs table.

Version 3.1 of this document applies the following modifications compared to version 3

- Further updates to references
- Several editorial corrections and clarifications
- Obsolete reference to RFC2616 and refer to the new set of RFCs for HTTP/1.1
- A clause is added on how to operate with transforming proxies and other adaptation middle-boxes in clause 3.4.4.
- Considerations on how to operate at the live edge for a live service
- The addition of the availability time offset to the description of the live service.
- The explicit exclusion of on-request based xlink for dynamic services.
- Clarifications of HEVC signaling for DASH in Table 18 based on feedback from MPEG.

- Clarify relation between SMPTE-TT and IMSC1 in clause 6.4.2 and 6.4.4.
- Add extension for audio codec AC-4.

Version 3.2 of this document applies the following modifications compared to version 3.1

- Further updates to references.
- Several editorial corrections and clarifications.
- Small clarification updates on the timing model in clause 3.2.7.
- Added support for switching across Adaptation Sets, in particular for H.264/AVC and H.265/HEVC switching in clause 3.8.
- Added a clarification on the value of `@audioSamplingRate` for AAC SBR in clause 6.3.2.
- Added a clause on the use of HTTPS with DASH in clause 7.2.
- Moved the test DRM to the test vector document 7.
- Added support for key rotation in clause 7.
- Add extension for MPEG-H audio in clause 9.

Version 3.3 of this document applies the following modifications compared to version 3.2

- Identifiers are summarized in the Introduction 1.
- References are updated
- Several editorial corrections and clarifications are applied
- Guidelines and permitted attributes when converting a live service to on-demand are documented, for details see clause 4.6.2.
- Added clarification on the format of remote elements in alignment with DCOR.3 of ISO/IEC 23009-1:2014 [4].
- Addressed the differentiation of different content types through mime types, codecs and Roles rather than the use `@contentType`. See clause 3.2.13 for more details.
- Added guidelines on how to use clear keys in the context of DASH, see clause 7.7.
- Provided guidelines on the timing model when using side-car subtitle files in DASH, see clause 6.4.4.
- Update period continuity to align with ISO/IEC 23009-1:2014/Amd.3 [4].
- Update the callback event to align with ISO/IEC 23009-1:2014/Amd.3 [4].
- Update the MPD anchors to align with ISO/IEC 23009-1:2014/Amd.3 [4].
- Take into account the updated in ISO/IEC 23009-1:2014/DCOR.3 [4].

Version 4.0 of this document applies the following modifications compared to version 3.3

- References are updated
- Several editorial corrections and clarifications are applied
- Update Adaptation Set Switching to align with ISO/IEC 23009-1:2014/Amd.3 [4].
- Add a recommendation on the setting of **`Period`**`@start` in case of multi-period content in clause 4.3.3.3.1.
- Add an external Representation Index for On-demand content in clause 3.2.1.

- Add Period connectivity to multi-period content to align with ISO/IEC 23009-1:2014/Amd.3 [4] in clause 3.2.12.
- Add clarifications and further recommendations and restriction to the use of the Time Synchronization in clause 4.7.
- Align Early Terminated Period with ISO/IEC 23009-1:2014/Amd.3 [4] in clause 4.8.3.
- Removal of interoperability requirements for CEA-708 in clause 6.4.3.4.
- References to EIDR and Ad-ID for Asset Identifiers in clause 5.5.1.
- Addition of W3C Clear Key with DASH in clause 7.6.2.4.
- Addition of guidelines for trick modes in live services in clause 4.10.
- Addition of UHD/HDR Extensions in clause 10.

Version 4.1 of this document applies the following modifications compared to version 4.0

- Several editorial fixes and updates.
- Guidelines for Robust Live Services in clause 4.11.
- Updates to W3C Clear Key usage in clause 7.6.2.4.
- Addition of last segment number signaling in clause 4.4.3.5 and 4.4.3.6.
- Addition of Dolby Vision dual layer in clause 10.4 and referenced Annexes
- Seek Preview and Thumbnail Navigation in clause 3.2.9 and clause 6.2.6
- Clarification on **`AudioChannelConfiguration`** element in clause 6.3.3.2
- Annotation and client model for content selection defined in clause 3.9.
- Remote referencing and use of query parameters for ad insertion in clause 5.3.2.1.3 and 5.3.5, respectively.
- Default KID Clarifications in clause 7.5.3.4.
- Addition of VP9 Video to DASH-IF IOP in clause 11.
- Updates to the references and separation of informative and normative references.
- Client and DRM System Interactions in clause 7 and in particular in clause 7.7.9

2.2.2. Backward-Compatibility Considerations

Generally, content can be offered such that it can be consumed by version 2 and version 3 clients. In such a case the restricted authoring should be used and it should be accepted that version 2 clients may ignore certain Representations and Adaptation Sets. Content Authors may also consider the publication of two MPDs, but use the same segment formats.

In terms of compatibility between version 2 and version 3, the following should be considered:

- The backward-compatibility across MPEG editions is handled in the second edition of ISO/IEC 23009-1 [4].
- General clarifications and updates are added
- Further restrictions on content authoring compared to version 2 are:
    - forbid time code wrap around
    - the usage of DRM, especially the Content Protection element
    - constraints on trick mode usage

    - additional constraints on the usage of HTTP
    - Adaptation Set constraints
- Relaxations are:
    - Permit usage of additional subtitling format based on CEA-608/708
    - the audio encoding requirements for HE-AACv2
    - permit to not use of the '`lmsg`' brand for signaling the last segment
    - the ability to signal bitstream switching set to true
    - the use of remote elements with Xlink

2.3. Use of Key Words

2.3.1. Background

DASH-IF generally does not write specifications, but provides and documents guidelines for implementers to refer to interoperability descriptions. In doing so, the DASH-IF agreed to use key words in order to support readers of the DASH-IF documents to understand better how to interpret the language. The usage of key words in this document is provided below.

2.3.2. Key Words

The key word usage is aligned with the definitions in RFC 2119 [44], namely:

- SHALL: This word means that the definition is an absolute requirement of the specification.
- SHALL NOT: This phrase means that the definition is an absolute prohibition of the specification.
- SHOULD: This word means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
- SHOULD NOT: This phrase means that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- MAY: This word means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item.

These key words are attempted to be used consistently in this document, but only in small letters.

2.3.3. Mapping to DASH-IF Assets

If an IOP document associates such a key word from above to a content authoring statement then the following applies:

- SHALL: The conformance software provides a conformance check for this and issues an *error* if the conformance is not fulfilled.
- SHALL NOT: The conformance software provides a conformance check for this and issues an *error* if the conformance is not fulfilled.

- SHOULD: The conformance software provides a conformance check for this and issues a *warning* if the conformance is not fulfilled.
- SHOULD NOT: The conformance software provides a conformance check for this and issues a *warning* if the conformance is not fulfilled.
- SHOULD and MAY: If present, the feature check of the conformance software documents a feature of the content.

If an IOP document associates such a key word from above to a DASH Client then the following applies:

- SHALL: Test content is necessarily provided with this rule and the reference client implements the feature.
- SHALL NOT: The reference client does not implement the feature.
- SHOULD: Test content is provided with this rule and the reference client implements the feature unless there is a justification for not implementing this.
- SHOULD NOT: The reference client does not implement the feature unless there is a justification for implementing this.
- MAY: Test content is provided and the reference client implements the feature if there is a justification this.

2.4. Definition and Usage of Interoperability Points

2.4.1. Profile Definition in ISO/IEC 23009-1

MPEG DASH defines formats for MPDs and Segments. In addition MPEG provides the ability to further restrict the applied formats by the definition of *Profiles* as defined on clause 8 of ISO/IEC 23009-1 [4]. Profiles of DASH are defined to enable interoperability and the signaling of the use of features.

Such a profile can also be understood as permission for DASH clients that implement the features required by the profile to process the Media Presentation (MPD document and Segments).

Furthermore, ISO/IEC 23009-1 permits external organizations or individuals to define restrictions, permissions and extensions by using this profile mechanism. It is recommended that such external definitions be not referred to as profiles, but as *Interoperability Points*. Such an interoperability point may be signalled in the `@profiles` parameter once a URI is defined. The owner of the URI is responsible to provide sufficient semantics on the restrictions and permission of this interoperability point.

This document makes use of this feature and provides a set of Interoperability Points. Therefore, based on the interoperability point definition, this document may be understood in two ways:

- a collection of content conforming points, i.e. as long as the content conforms to the restrictions as specified by the IOP, clients implementing the features can consume the content.
- a client capability points that enable content and service providers for flexible service provisioning to clients conforming to these client capabilities.

This document provides explicit requirements, recommendations and guidelines for content authoring that claims conformance to a profile (by adding the `@profiles` attribute to the MPD) as well as for clients that are permitted to consume a media presentation that contains such a profile.

2.4.2. Usage of Profiles

A Media Presentation may conform to one or multiple profiles/interoperability points and conforms to each of the profiles indicated in the **MPD**`@profiles` attribute is specified as follows:

When `ProfA` is included in the **MPD**`@profiles` attribute, the MPD is modified into a profile-specific MPD for profile conformance checking using the following ordered steps:

1. The **MPD**`@profiles` attribute of the profile-specific MPD contains only `ProfA`.
2. An **AdaptationSet** element for which `@profiles` does not or is not inferred to include `ProfA` is removed from the profile-specific MPD.
3. A Representation element for which `@profiles` does not or is not inferred to include `ProfA` is removed from the profile-specific MPD.
4. All elements or attributes that are either (i) in this Part of ISO/IEC 23009 and explicitly excluded by `ProfA`, or (ii) in an extension namespace and not explicitly included by `ProfA`, are removed from the profile-specific MPD.
5. All elements and attributes that "may be ignored" according to the specification of `ProfA` are removed from the profile-specific MPD.

An MPD is conforming to profile `ProfA` when it satisfies the following:

1. `ProfA` is included in the **MPD**`@profiles` attribute.
2. The profile-specific MPD for `ProfA` conforms to ISO/IEC 23009-1
3. The profile-specific MPD for `ProfA` conforms to the restrictions specified for `ProfA`.

A Media Presentation is conforming to profile `ProfA` when it satisfies the following:

1. The MPD of the Media Presentation is conforming to profile `ProfA` as specified above.
2. There is at least one Representation in each Period in the profile-specific MPD for `ProfA`.
3. The Segments of the Representations of the profile-specific MPD for `ProfA` conform to the restrictions specified for `ProfA`.

2.4.3. Interoperability Points and Extensions

This document defines Interoperability Points and Extensions. Both concepts make use of the profile functionality of ISO/IEC 23009-1.

Interoperability Points provide a basic collection of tools and features to ensure that content/service providers and client vendors can rely to support a sufficiently good audio-visual experience. Extensions enable content/service providers and client vendors to enhance the audio-visual experience provided by an Interoperability Point in a conforming manner.

The only difference between Interoperability Points and Extensions is that Interoperability Points define a full audio-visual experience and Extensions enhance the audio-visual experience in typically only one dimension.

Examples for the usage of the `@profiles` signaling are provided in Annex A of this document.

3. DASH-Related Aspects

3.1. Scope

DASH-IF Interoperability Points use ISO base media file format [7] based encapsulation and provide significant commonality with a superset of the ISO BMFF On-Demand, the ISO BMFF Live profile, and the ISO BMFF Common Profile as defined in ISO/IEC 23009-1 [4], sections 8.3, 8.4 and 8.10, respectively. DASH-IF IOPs are intended to provide support for on-demand and live content. The primary constraints imposed by this profile are the requirement that each Representation is provided in one of the following two ways

- as a single Segment, where Subsegments are aligned across Representations within an Adaptation Set. This permits scalable and efficient use of HTTP servers and simplifies seamless switching. This is mainly for on-demand use cases.
- as a sequence of Segments where each Segment is addressable by a template-generated URL. Content generated in this way is mainly suitable for dynamic and live services.

In both cases (Sub)Segments begin with Stream Access Points (SAPs) of type 1 or 2 [7], i.e. regular IDR frames in case of video. In addition, (Sub)Segments are constrained so that for switching video Representations within one Adaptation Set the boundaries are aligned without gaps or overlaps in the media data. Furthermore, switching is possible by a DASH client that downloads, decodes and presents the media stream of the come-from Representation and then switches to the go-to Representation by downloading, decoding and presenting the new media stream. No overlap in downloading, decoding and presentation is required for seamless switching of Representations in one Adaptation Set.

Additional constraints are documented for bitstream switching set to true as well as special case such as trick modes, etc.

3.2. DASH Formats

3.2.1. Introduction

This section introduces the detailed constraints of the MPD and the DASH segments in a descriptive way referring to ISO/IEC 23009-1 [4]. The DASH-based restrictions have significant commonality with the ISO BMFF Live and On-Demand profiles from the MPEG-DASH specification.

Specifically:

- Segment formats are based on ISO BMFF with fragmented movie files, i.e. (Sub)Segments are encoded as movie fragments containing a track fragment as defined in ISO/IEC 14496-12 [7], plus the following constraints to make each movie fragment independently decodable:
  - Default parameters and flags shall be stored in movie fragments (‘`tfhd`’ or ‘`trun`’ box) and not track headers (‘`trex`’ box)
  - The ‘`moof`’ boxes shall not use external data references, the flag ‘`default-base-is-moof`’ shall also be set (aka movie-fragment relative addressing) and `data-offset` shall be used, i.e. `base-data-offset-present` shall not be used (follows ISO/IEC 23009-1 [4]).

- Alignment with ISO BMFF Live & On-Demand Profiles, i.e. within each Adaptation Set the following applies
    - Fragmented movie files are used for encapsulation of media data
    - (Sub)Segments are aligned to enable seamless switching

Beyond the constraints provided in the ISO BMFF profiles, the following additional restrictions are applied.

- IDR-like SAPs (i.e., SAPs type 2 or below) at the start of each (Sub)Segment for simple switching.
- Segments should have almost equal duration. The maximum tolerance of segment duration shall be ±50% and the maximum accumulated deviation over multiple segments shall be ±50% of the signaled segment duration (i.e. the `@duration`). Such fluctuations in actual segment duration may be caused by for example ad replacement or specific IDR frame placement. Note that the last segment in a Representation may be shorter according to ISO/IEC 23009-1 [4].

    Note 1: If accurate seeking to specific time is required and at the same time a fast response is required one may use On-Demand profile for VoD or the **`SegmentTimeline`** based addressing. Otherwise the offset in segment duration compared to the actual media segment duration may result in a less accurate seek position for the download request, resulting in some increased initial start-up.

    Note 2: The maximum tolerance of segment duration is also addressed in ISO/IEC 23009-1:2014/Cor.3 [4] and once approved, a reference to the specification may replace the above text.

- If the **`SegmentTimeline`** element is used for the signaling of the Segment duration, the timing in the segment timeline shall be media time accurate and no constraints on segment duration deviation are added except the maximum segment duration as specified in the MPD. However, despite the usage of the the **`SegmentTimeline`**, it is not encouraged to use varying Segment durations. The **`SegmentTimeline`** element should only be used in order to signal occasional shorter Segments (possibly caused by encoder processes) or to signal gaps in the time line.
- only non-multiplexed Representations shall be used, i.e. each Representation only contains a single media component.
- Addressing schemes are restricted to
    - templates with number-based addressing
    - templates with time-based addressing
    - For on-demand profiles, the Indexed Media Segment as defined in ISO/IEC 23009-1 [4], clause 6.3.4.4: In this case either the `@indexRange` attribute shall be present or the **`RepresentationIndex`** element shall be present. Only a single `sidx` box shall be present.

        Note 1: the external Representation Index was only added in the second edition [4]. If compatibility to v2.0 or earlier of this document is necessary, the external Representation Index shall not be used.

Note 2: The latter restriction was introduced in version 3 of this document based on deployment experience and to enable alignment with DVB DASH.

- In case multiple Video Adaptation Sets as defined in 3.2.13 are offered, exactly one video Adaptation Set shall be signaled as the main one unless different Adaptation Sets contain the same content with different quality or different codecs. In the latter case, all Adaptation Sets with the same content shall be signaled as the main content. Signaling as main content shall be done by using the Role descriptor with @schemeIdUri=" urn:mpeg:dash:role:2011" and @value="main".
- The content offering shall adhere to the presence rules of certain elements and attributes as defined section 3.2.4.

It is expected that a client conforming to such a profile is able to process content offered under these constraints. More details on client procedures are provided in section 3.3.

3.2.2. Media Presentation Description constraints for v1 & v2 Clients

3.2.2.1. Definition according to ISO/IEC 23009-1

This section follows a description according to ISO/IEC 23009-1. In section 3.2.2.2, a restricted content offering is provided that provides a conforming offering.

NOTE: The term "ignored" in the following description means, that if an MPD is provided and a client that complies with this interoperability point removes the element that may be ignored, then the MPD is still complying with the constraints of the MPD and segments as defined in ISO/IEC 23001-9, section 7.3.

The MPD shall conform to the ISO Base Media File Format Common profile as defined on ISO/IEC 23009-1:2014/Amd.1:2015 [4], section 8.9, except for the following issues:

- Representations with @mimeType attribute application/xml+ttml shall not be ignored.

In addition, the Media Presentation Description shall conform to the following constraints:

— **Representation** elements with a @subsegmentStartsWithSAP value set to 3 may be ignored.

— **Representation** elements with a @startsWithSAP value set to 3 may be ignored.

— If a Period contains multiple Video Adaptation Sets as defined in 3.2.13 then at least one Adaptation Set shall contain a Role element <Role schemeIdUri="urn:mpeg:dash:role:2011" value="main"> and each Adaptation Set containing such a **Role** element shall provide perceptually equivalent media streams.

3.2.2.2. Simple Restricted Content Offering

A conforming MPD offering based on the ISO BMFF Live Profile shall contain

- **MPD**@type set to static or set to dynamic.
- **MPD**@profiles includes urn:mpeg:dash:profile:isoff-live:2011
- One or multiple Periods with each containing one or multiple Adaptation Sets and with each containing one or multiple Representations.

- The Representations contain or inherit a **SegmentTemplate** with `$Number$` or `$Time$` Identifier.
- `@segmentAlignment` set to true for all Adaptation Sets

A conforming MPD offering based on the ISO BMFF On-Demand Profile shall contain

- **MPD**`@type` set to `static`.
- **MPD**`@profiles` includes `urn:mpeg:dash:profile:isoff-ondemand:2011`
- One or multiple Periods with each containing one or multiple Adaptation Sets and with each containing one or multiple Representations.
- `@subSegmentAlignment` set to true for all Adaptation Sets

3.2.3. Segment format constraints

Representations and Segments referred to by the Representations in the profile-specific MPD for this profile, the following constraints shall be met:

— Representations shall comply with the formats defined in ISO/IEC 23009-1, section 7.3.

— In Media Segments, all Segment Index ('`sidx`') and Subsegment Index ('`ssix`') boxes, if present, shall be placed before any Movie Fragment ('`moof`') boxes.

Note: DVB DASH [42] permits only one single Segment Index box ('sidx') for the entire Segment. As this constraints is not severe in the content offering, it is strongly recommended to offer content following this constraint.

— If the **MPD**`@type` is equal to "`static`" and the **MPD**`@profiles` attribute includes "`urn:mpeg:dash:profile:isoff-on-demand:2011`", then

  — Each Representation shall have one Segment that complies with the Indexed Self-Initializing Media Segment as defined in section 6.3.5.2 in ISO/IEC 23009-1.

— Time Codes expressing presentation and decode times shall be linearly increasing with increasing Segment number in one Representation. In order to minimize the frequency of time code wrap around 64 bit codes may be used or the timescale of the Representation may be chosen as small as possible. In order to support time code wrap around, a new Period may be added in the MPD added that initiates a new Period in order to express a discontinuity.

3.2.4. Presence of Attributes and Elements

Elements and attributes are expected to be present for certain Adaptation Sets and Representations to enable suitable initial selection and switching. Simple rules are provided in this section. A detailed description of multi-track content offering is provided in clause 3.9.

Specifically the following applies:

- For any Video Adaptation Sets as defined in 3.2.13 the following attributes shall be present
  - `@maxWidth` (or `@width` if all Representations have the same width)

- `@maxHeight` (or `@height` if all Representations have the same height)
- `@maxFrameRate` (or `@frameRate` if all Representations have the same frame rate)
- `@par`

Note: The attributes `@maxWidth` and `@maxHeight` should be used such that they describe the target display size. This means that they may exceed the actual largest size of any coded Representation in one Adaptation Set.

- The attributes `@minWidth` and `@minHeight` should not be present. If present, they may be smaller than the smallest `@width` or smallest `@height` in the Adaptation Set.

- For any Representation within a Video Adaptation Sets as defined in 3.2.13 the following attributes shall be present:
  - `@width`, if not present in **`AdaptationSet`** element
  - `@height`, if not present in **`AdaptationSet`** element
  - `@frameRate`, if not present in **`AdaptationSet`** element
  - `@sar`

  Note: `@width`, `@height`, and `@sar` attributes should indicate the vertical and horizontal sample count of encoded and cropped video samples, not the intended display size in pixels.
- For Adaptation Set or for any Representation within an Video Adaptation Sets as defined in 3.2.13 the attribute `@scanType` shall either not be present or shall be set to "`progressive`".
- For any Audio Adaptation Sets as defined in 3.2.13 the following attributes shall be present
  - `@lang`
- For any Representation within an Audio Adaptation Sets as defined in 3.2.13 the following elements and attributes shall be present:
  - `@audioSamplingRate`, if not present in **`AdaptationSet`** element
  - **`AudioChannelConfiguration`**, if not present in **`AdaptationSet`** element

3.2.5. MPD Dimension Constraints

No constraints are defined on MPD size, or on the number of elements. However, it should be avoided to create unnecessary large MPDs.

Note: DVB DASH [42] adds MPD dimension constraints in section 4.5 of their specification. In order to conform to this specification, it is recommended to obey these constraints.

3.2.6. Generic Metadata

Generic metadata may be added to MPDs based on DASH. For this purpose, the Essential Property Descriptor and the Supplemental Property Descriptor as defined in ISO/IEC 23009-1 [4], clause 5.8.4.7 and 5.8.4.8.

Metadata identifiers for content properties are provided here: `http://dashif.org/identifiers`.

However, it is not expected that DASH-IF clients support all metadata at `http://dashif.org/identifiers` unless explicitly required.

3.2.7. DASH Timing Model

3.2.7.1. General

According to ISO/IEC 23009-1, DASH defines different timelines. One of the key features in DASH is that encoded versions of different media content components share a common timeline. The presentation time of each access unit within the media content is mapped to the global common presentation timeline for synchronization of different media components and to enable seamless switching of different coded versions of the same media components. This timeline is referred as Media Presentation timeline. The Media Segments themselves contain accurate Media Presentation timing information enabling synchronization of components and seamless switching.

A second timeline is used to signal to clients the availability time of Segments at the specified HTTP-URLs. These times are referred to as **Segment availability times** and are provided in wall-clock time. Clients typically compare the wall-clock time to Segment availability times before accessing the Segments at the specified HTTP-URLs in order to avoid erroneous HTTP request responses. For static Media Presentations, the availability times of all Segments are identical. For dynamic Media Presentations, the availability times of segments depend on the position of the Segment in the Media Presentation timeline, i.e. the Segments get available and unavailable over time.

Figure 1 provides an overview of the different timelines in DASH and their relation. The diagram shows three Periods, each of the Periods contains multiple Representations (for the discussion it is irrelevant whether these are included in the same Adaptation Set or in different ones).

Specifically, the following information is available in the MPD that relates to timing:

- **`MPD`**`@availabilityStartTime`: the start time is the anchor for the MPD in wall-clock time. The value is denoted as *AST*.
- **`Period`**`@start`: the start time of the Period relative to the MPD availability start time. The value is denoted as *PS*.
- **`Representation`**`@presentationTimeOffset`: the presentation time offset of the Representation in the Period, i.e. it provides the media time of the Representation that is supposed to be rendered at the start of the Period. Note that typically this time is either earliest presentation time of the first segment or a value slightly larger in order to ensure synchronization of different media components. If larger, this Representation is presented with short delay with respect to the Period start.

In addition, with the use of the **`Representation`**`@duration` or **`Representation.SegmentTimeline`** the MPD start time for each segment and the MPD duration for each segment can be derived. For details refer to ISO/IEC 23009-1 as well as section 4.

According to Figure 1, the *AST* is a wall-clock time. It provides an anchor to all wall-clock time computation in the MPD. The sum of the **`Period`**`@start` of the first Period and the *AST* provides the Period start time *PST1* value in wall-clock time of the first Period.

The media timeline origin for tracks in ISO Media files is defined to be zero. Each Representation is assigned a presentation time offset, either by the value of the attribute **Representation**@presentationTimeOffset or by default set to 0. The value of this attribute is denoted as PTO. It is normally the case that for complete files sequenced as Periods this value is 0. For "partial" files or live streams, @presentationTimeOffset indicates the media composition/presentation time of the samples that synchronize to the start of the Period. @presentationTimeOffset for the live stream will usually not be zero because the encoders are usually started prior to presentation availability, so the media timestamps on the first available Segments will have increased since the encoders started. Encoding timestamps may be set to UTC (as though the encoder was turned on 1/1/1970 at midnight). Representations in Periods of live content typically have the same @presentationTimeOffset as long as the media is continuously encoded, because UTC time and media time increase at the same rate and maintain the same offset.

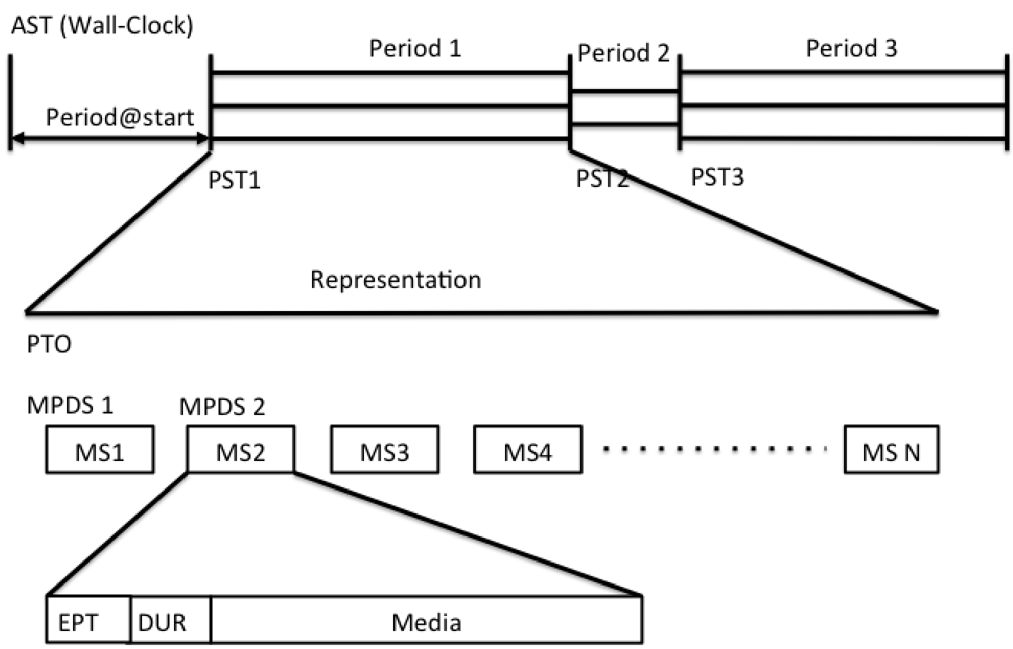

Figure 1 Overview Timing Model

Within a Representation, each Segment is assigned an MPD start time and MPD duration according to ISO/IEC 23009-1 (more details for dynamic services are provided in section 4). These two values can be computed from the MPD and provide approximate times for each segment that are in particular useful for random access and seeking.

In addition, each segment has an internal sample-accurate presentation time. Therefore, each segment has a media internal earliest presentation time *EPT* and sample-accurate presentation duration *DUR*.

For each media segment in each Representation the MPD start time of the segment should approximately be *EPT - PTO*. Specifically, the MPD start time shall be in the range of *EPT - PTO - 0.5*DUR* and *EPT - PTO + 0.5*DUR* according to the requirement stated above.

Each Period is treated independently. Details on processing at Period boundaries are provided in ISO/IEC 23009-1. One example is, that for time code wrap-around a new Period is expected to be added that restarts at presentation time 0.

The value of **Period**@start for an ad can be chosen to coincide with an insertion point in the live stream by setting **Period**@start to a presentation time duration equal to the UTC time difference between @availabilityStartTime and the scheduled encoding time of the insertion point in the live stream.

3.2.7.2. Static Media Presentations

For static media presentations, all Segments shall be available at time *AST*. This means that a DASH client may use the information in the MPD in order to seek to approximate times.

3.2.7.3. Dynamic Media Presentations

For dynamic media presentations, segments get available over time. The latest time they shall be available is at the sum of PST (which is *AST* + **Period**@start), MPD start time and MPD duration. The latter is added in order to take into account that at the server a segment typically needs to be completed prior to its availability. For details refer to section 4.

3.2.8. Bandwidth and Minimum Buffer Time

The MPD contains a pair of values for a bandwidth and buffering description, namely the Minimum Buffer Time (MBT) expressed by the value of **MPD**@minBufferTime and bandwidth (BW) expressed by the value of **Representation**@bandwidth. The following holds:

- the value of the minimum buffer time does not provide any instructions to the client on how long to buffer the media. The value however describes how much buffer a client should have under ***ideal*** network conditions. As such, MBT is not describing the burstiness or jitter in the network, it is describing the burstiness or jitter in the **content encoding**. Together with the BW value, it is a property of the content. Using the "leaky bucket" model, it is the size of the bucket that makes BW true, given the way the content is encoded.
- The minimum buffer time provides information that for each Stream Access Point (and in the case of DASH-IF therefore each start of the Media Segment), the property of the stream: If the Representation (starting at any segment) is delivered over a constant bitrate channel with bitrate equal to value of the BW attribute then each presentation time *PT* is available at the client latest at time with a delay of at most *PT* + *MBT*.
- In the absence of any other guidance, **the** MBT **should be set** to the maximum GOP size (coded video sequence) of the content, which quite often is identical **to the maximum segment duration** for the live profile or the **maximum subsegment duration** for the On-Demand profile. The *MBT* may be set to a smaller value than maximum (sub)segment duration, but should not be set to a higher value.

In a simple and straightforward implementation, a DASH client decides downloading the next segment based on the following status information:

- the currently available buffer in the media pipeline, *buffer*
- the currently estimated download rate, *rate*
- the value of the attribute @minBufferTime, *MBT*
- the set of values of the @bandwidth attribute for each Representation i, *BW*[*i*]

The task of the client is to select a suitable Representation *i*.

The relevant issue is that starting from a SAP on, the DASH client can continue to playout the data. This means that at the current time it does have *buffer* data in the buffer. Based on this model the client can download a Representation *i* for which $BW[i] \leq rate*buffer/MBT$ without emptying the buffer.

Note that in this model, some idealizations typically do not hold in practice, such as constant bitrate channel, progressive download and playout of Segments, no blocking and congestion of other HTTP requests, etc. Therefore, a DASH client should use these values with care to compensate such practical circumstances; especially variations in download speed, latency, jitter, scheduling of requests of media components, as well as to address other practical circumstances.

One example is if the DASH client operates on Segment granularity. As in this case, not only parts of the Segment (i.e., `MBT`) needs to be downloaded, but the entire Segment, and if the `MBT` is smaller than the Segment duration, then rather the segment duration needs to be used instead of the `MBT` for the required buffer size and the download scheduling, i.e. download a Representation *i* for which $BW[i] \leq rate*buffer/max\_segment\_duration$.

3.2.9. Trick Mode Support

Trick Modes are used by DASH clients in order to support fast forward, seek, rewind and other operations in which typically the media, especially video, is displayed in a speed other than the normal playout speed. In order to support such operations, it is recommended that the content author adds Representations at lower frame rates in order to support faster playout with the same decoding and rendering capabilities.

However, Representations targeted for trick modes are typically not be suitable for regular playout. If the content author wants to explicitly signal that a Representation is only suitable for trick mode cases, but not for regular playout, the following is recommended:

- add one or multiple Adaptation Sets that that only contains trick modes Representations
- annotate each Adaptation Set with an **`EssentialProperty`** descriptor or **`SupplementalProperty`** descriptor with URL `"http://dashif.org/guidelines/trickmode"` and the `@value` the value of `@id` attribute of the Adaptation Set to which these trick mode Representations belong. The trick mode Representations must be time-aligned with the Representations in the main Adaptation Set. The value may also be a white-space separated list of @id values. In this case the trick mode Adaptation Set is associated to all Adaptation Sets with the values of the `@id`.
- signal the playout capabilities with the attribute `@maxPlayoutRate` for each Representation in order to indicate the accelerated playout that is enabled by the signaled codec profile and level.
- If the Representation is encoded without any coding dependency on the elementary stream level, i.e. each sample is a SAP type 1, then it is recommended to set the `@codingDependency` attribute to `FALSE`.
- If multiple trick mode Adaptation Sets are present for one main Adaptation Set, then sufficient signaling should be provided to differentiate the different trick mode Adaptation Sets. Different Adaptation Sets for example may be provided as thumbnails (low spatial resolution), for fast forward or rewind (no coding dependency with `@codingDependency` set to false and/or lower frame rates), longer values for `@duration` to improve download frequencies or different `@maxPlayoutRate` values. Note also that the `@bandwidth` value should be carefully documented to support faster than real-time download of Segments.

If an Adaptation Set in annotated with the **EssentialProperty** descriptor with URI "`http://dashif.org/guidelines/trickmode` then the DASH client shall not select any of the contained Representations for regular playout.

For trick modes for live services, the same annotation should be used. More details on service offerings are provided in section 4.10.

3.2.10. Adaptation Set Constraints

3.2.10.1. Introduction

Content in one Adaptation Set is constrained to enable and simplify switching across different Representations of the same source content. General Adaptation Set constraints allow sequencing of Media Segments from different Representations ("bitrate switching") prior to a single audio or video decoder, typically requiring the video decoder to be reset to new decoding parameters at the switch point, such as a different encoded resolution or codec profile and level.

Bitstream Switching Adaptation Set constraints allow a switched sequence of Media Segments to be decoded without resetting the decoder at switch points because the resulting Segment stream is a valid track of the source type, so the decoder is not even aware of the switch. In order to signal that the Representations in an Adaptation Set are offered under these constraints, the attribute **AdaptationSet**`@bitstreamSwitching` may be set to `true`. When **AdaptationSet**`@bitstreamSwitching` attribute is set to `TRUE`, the decoder can continue decoding without re-initialization. When `@bitstreamSwitching` is set to **FALSE**, seamless switching across Representations can be achieved with re-initialization of the decoder. Content authors should set **AdaptationSet**`@bitstreamSwitching` to **TRUE** only if the content does not need the decoder to be re-initialized.

In the following general requirements and recommendations are provided for content in an Adaptation Set in section 3.2.10.2 and specific requirements when the bitstream switching is set to true in section 3.2.10.3.

3.2.10.2. General

General Adaptation Set constraints require a client to process an Initialization Segment prior to the first Media Segment and prior to each Media Segment selected from a different Representation (a "bitrate switch").

Adaptation Sets shall contain Media Segments compatible with a single decoder that start with SAP type 1 or 2, and in time aligned Representations using the same `@timescale`, when multiple Representations are present.

Edit lists in Initialization Segments intended to synchronize the presentation time of audio and video should be identical for all Representations in an Adaptation Set.

Note: Re-initialization of decoders, decryptors, and display processors on some clients during bitrate switches may result in visible or audible artifacts. Other clients may evaluate the differences between Initialization Segments to minimize decoder reconfiguration and maintain seamless presentation equal to the encoded quality.

Additional recommendations and constraints may apply for encryption and media coding. For details, please check the relevant sections in this document, in particular section 6.2.5 and 7.7.5.

3.2.10.3. Bitstream Switching

A bitstream switching Adaptation Set is optimized for seamless decoding, and live streams that may change encoding parameters over time. A bitstream switching Adaptation Set may process an Initialization Segment one time from the highest bandwidth Representation in the Adaptation Set, and then process Media Segments from any other Representation in the same Adaptation Set without processing another Initialization Segment. The resulting sequence of an Initialization Segment followed by time sequenced Media Segments results in a valid ISO BMFF file with an elementary stream similar to a transport stream.

For all Representations within an Adaptation Set with `@bitstreamSwitching='true'`:

- the `Track_ID` shall be equal for all Representations
- Each movie fragment shall contain one track fragment

Note: Multiple Adaptation Sets may be included in an MPD that contain different subsets of the available Representations that are optimized for different decoder and screen limitations. A Representation may be present in more than one Adaptation set, for example a 720p Representation that is present in a 720p Adaptation Set may also be present in a 1080p Adaptation Set. The 720p Representation uses the same Initialization Segments in each Adaptation Set, but the 1080p Adaptation Set would require decoder and display configuration with the 1080p Initialization Segment.

Additional recommendation and constraints may apply for encryption and media coding. For details, please see below.

3.2.11. Media Time Information of Segment

The earliest presentation time may be estimated from the MPD using the segment availability start time minus the segment duration announced in the MPD.

The earliest presentation time may be accurately determined from the Segment itself.

If the Segment Index is present than this time is provided in the `earliest_presentation_time` field of the Segment Index. To determine the presentation time in the Period, the value of the attribute `@presentationTimeOffset` needs to be deducted.

If the Segment Index is not present, then the earliest presentation time is deduced from the ISO BMFF parameters, namely the movie fragment header and possibly in combination with the information in the Initialization Segment using the edit list.

The earliest presentation time in the Period for a Segment can be deduced from the decode time taking also into account the composition time offset, edit lists as well as presentation time offsets.

Specifically the following is the case to determine the earliest presentation time assuming that no edit list is present in the Initialization Segment:

- If the SAP type is 1, then the earliest presentation time is identical to the sum of the decode time and the composition offset of the first sample. The decode time of the first sample is determined by the base media decode time of the movie fragment.
- If the SAP type is 2, the first sample may not be the sample with the earliest presentation time. In order to determine the sample with the earliest presentation time, *this sample* is determined as the sample for which the sum of the decode time and the composition offset is the smallest within this Segment. Then the earliest presentation time of the Segment is the sum of the base media decode time and the sum of the decode time and the composition offset for *this sample*. Such an example is shown below.

In addition, if the presentation time needs to be adjusted at the beginning of a period, then the `@presentationTimeOffset` shall be used in order to set the presentation that is mapped to the start of the period. Content authoring shall be such that if edit lists are ignored, then the client can operate without timing and lip sync issues.In the following examples, there is a sequence of I, P, and B frames, each with a decoding time delta of 10. The segmentation, presentation order and storage of the samples is shown in the table below. The samples are stored with the indicated values for their decoding time deltas and composition time offsets (the actual CT and DT are given for reference). The re-ordering occurs because the predicted P frames must be decoded before the bi-directionally predicted B frames. The value of DT for a sample is always the sum of the deltas of the preceding samples. Note that the total of the decoding deltas is the duration of the media in this track.

Example with closed GOP and SAP Type = 1:

Segment	/-	--	--	--	--	--	--	/-	--	--	--	--	--	--\
	I1	P4	B2	B3	P7	B5	B6	I8	P1	B9	B1	P1	B1	B13
Presenta-tion Order	|==| I1 B2 B3 P4 B5 B6 P7 |==| I8 B9 B10 P11 B12 B13 P14 |==|													
Base media decode time	0							70						
Decode delta	10	10	10	10	10	10	10	10	10	10	10	10	10	10
DT	0	10	20	30	40	50	60	70	80	90	10	11	12	130
EPT	10							80						
Composition time offset	10	30	0	0	30	0	0	10	30	0	0	30	0	0
CT	10	40	20	30	70	50	60	80	11	90	10	14	12	130

Example with closed GOP and SAP Type = 2:

Segment	/--	--	--	--	--	--\	/-	--	--	--	---	--\
	I3	P1	P2	P6	B4	B5	I9	P7	P8	P12	B10	B11
Presenta-tion Order	|==| P1 P2 I3 B4 B5 P6 |==| P7 P8 I9 B10 B11 P12 |==|											
Base media decode time	0						60					
Decode Delta	10	10	10	10	10	10	10	10	10	10	10	10
DT	0	10	20	30	40	50	60	70	80	90	100	110
EPT	10						70					
Composition time offset	30	0	0	30	0	0	30	0	0	30	0	0

CT	30	10	20	60	40	50	90	70	80	120	100	110

Example with closed GOP and SAP Type = 2 and negative composition offset:

Segment	/--	--	--	--	--	--\	/-	--	--	--	---	--\
	I3	P1	P2	P6	B4	B5	I9	P7	P8	P12	B10	B11
Presentation Order	|==| P1 P2 I3 B4 B5 P6 |==| P7 P8 I9 B10 B11 P12 |==|											
Base media decode time	0						60					
Decode Delta	10	10	10	10	10	10	10	10	10	10	10	10
DT	0	10	20	30	40	50	60	70	80	90	100	110
EPT	0						60					
Composition offset	20	-10	-10	20	-10	-10	20	-10	-10	20	-10	-10
CT	20	0	10	50	30	40	80	60	70	110	90	100

For additional details refer to ISO/IEC 14496-12 [7] and ISO/IEC 23009-1 [1].

3.2.12. Content Offering with Periods

Content may be offered with a single Period. If content is offered with a single Period it is suitable to set PSTART to zero, i.e. the initialization segments get available at START on the server. However, other values for PSTART may be chosen.

> Note: This is aligned with Amd.3 of ISO/IEC 23009-1:2014 [4] and may be referenced in a future version of this document.

Content with multiple Periods may be created for different reasons, for example:

- to enable splicing of content, for example for ad insertion,
- to provide synchronization in segment numbering, e.g. compensate non-constant segment durations
- to remove or add certain Representations in an Adaptation Set,
- to remove or add certain Adaptation Sets,
- to remove or add content offering on certain CDNs,
- to enable signalling of shorter segments, if produced by the encoder.
- for robustness reasons as documented in detail in section 4.8.

Periods provide opportunities for resync, for ad insertion, for adding and removing Representations. However, in certain circumstances the content across Period boundaries is continuous and in this case, continuous playout of the client is expected.

In certain circumstances the Media Presentation is offered such that the next Period is a continuation of the content in the previous Period, possibly in the immediately following Period or in a later Period (e.g after an advertisement Period had been inserted), in particular that certain media components are continued.

The content provider may express that the media components contained in two Adaptation Sets in two different Periods are associated by assigning equivalent Asset Identifiers to both Periods and by identifying both Adaptation Sets with identical value for the attribute `@id`. Association expresses a logical continuation of the media component in the next Period and may for example be used by the client to continue playing an associated Adaptation Set in the new Period.

In addition, two Adaptation Sets in one MPD are period-continuous if all of the following holds:

- The Adaptation Sets are associated.
- The sum of the value of the `@presentationTimeOffset` and the presentation duration of all Representations in one Adaptation Set are identical to the value of the @presentationTimeOffset of the associated Adaptation Set in the next Period.
- If Representations in both Adaptation Sets have the same value for `@id`, then they shall have functionally equivalent Initialization Segments, i.e. the Initialization Segment may be used to continue the play-out of the Representation. The concatenation of the Initialization Segment of the first Period, if present, and all consecutive Media Segments in the Representation in the first Period and subsequently the concatenation with all consecutive Media Segments in the Representation of the second Period shall represent a conforming Segment sequence as defined in 4.5.4 conforming to the media type as specified in the `@mimeType` attribute for the Representation in the first Period. Additionally, the `@mimeType` attribute for the Representation in the next Period shall be the same as one of the first Period.

Media Presentations should signal period-continuous Adaptation Sets by using a supplemental descriptor on Adaptation Set level with `@schemeIdUri` set to `"urn:mpeg:dash:period-continuity:2015"` with

- the `@value` of the descriptor matching the value of an `@id` of a Period that is contained in the MPD,
- the value of the **`AdaptationSet`**`@id` being the same in both Periods.

MPD should signal period-continuous Adaptation Sets if the MPD contains Periods with identical Asset Identifiers.

There exist special cases, for which the media in one Adaptation Set is a continuation of the previous one, but the timestamps are not continuous. Examples are timestamp wrap around, encoder reset, splicing, or other aspects. Two Adaptation Sets in one MPD are period-connected if all conditions from period-continuity from above hold, except that the timestamps across Period boundaries may be non-continuous, but adjusted by the value of the `@presentationTimeOffset` at the Period boundary. However, for example the Initialization Segment is equivalent within the two Adaptation Sets. Media Presentations should signal period-connected Adaptation Sets by using a supplemental descriptor on Adaptation Set level with `@schemeIdUri` set to `"urn:mpeg:dash:period-connectivity:2015"`.

Note that period continuity implies period connectivity.

The content author should use period-continuity signaling or period-connectivity signaling if the content follows the rules. The client should exploit such signals for seamless user experience across Period boundaries.

For details on content offering with multiple Periods, please refer to the requirements and recommendations in section 4 and 5.

3.2.13. Adaptation Set Media Type

In contrast to MPEG-DASH which does not prohibit the use of multiplexed Representations, in the DASH-IF IOPs one Adaptation Set always contains exactly a single media type. The following media types for Adaptation Sets are defined:

- Video Adaptation Set: An Adaptation Set that contains visual information for display to the user. Such an Adaptation Set is identified by `@mimeType="video/mp4"`. For more details on the definition of media type video, refer to RFC 4337 [66]. The DASH-IF IOP restrict the usage of `video/mp4` to only `@codecs` values as defined in this specification.
- Audio Adaptation Set: An Adaptation Set that contains sound information to be rendered to the user. Such an Adaptation Set is identified by `@mimeType="audio/mp4".` For more details on the definition of media type video, refer to RFC 4337 [66]. The DASH-IF IOP restrict the usage of `audio/mp4` to only `@codecs` values as defined in this specification.
- Subtitle Adaptation Set: An Adaptation Set that contains visual overlay information to be rendered as auxiliary or accessibility information. Such an Adaptation Set is identified by `@mimeType="application/mp4"`, a Role descriptor with `@schemeIdUri=" urn:mpeg:dash:role:2011"` and `@value="subtitle"` and, an `@codecs` parameter as defined in Table 21or `"application/ttml+xml"`.
- Metadata Adaptation Set: An Adaptation Set that contains information that is not expected to be rendered by a specific media handler, but is interpreted by the application. Such an Adaptation Set is identified by `@mimeType="application/mp4"` and an appropriate sample entry identified by the `@codecs` parameter.

The media type is used by the DASH client in order to identify the appropriate handler for rendering. Typically, the DASH client selects at most one Adaptation Set per media type. In addition, the DASH client uses the string included in the `@codecs` parameter in order to identify if the underlying media playback platform can play the media contained in the Representation.

3.2.14. Seek Preview and Thumbnail Navigation

Seek preview and thumbnail navigation provide DASH clients the possibility to implement thumbnails for UI scrubbing. This may be implemented using a separate video Adaptaion Set and using trick mode features as defined in clause 3.2.9. However, this feature may be relatively complex to implement in a player and requires double video decoders. In a simpler approach, a sequence of image tiles may be used, each with multiple thumbnails to provide such thumbnails. An interoperable solution is provided in clause 6.2.6.

3.3. Client Implementation Requirements and Guidelines

3.3.1. Overview

The DASH-related aspects of the interoperability point as defined in section 3.2 can also be understood as permission for DASH clients that only implement the features required by the description to process the Media Presentation (MPD document and Segments). The detailed DASH-related client operations are not specified. Therefore, it is also unspecified how a DASH client exactly conforms. This document however provides guidelines on what is expected for conformance to this interoperability point. A minimum set of requirements is collected in section 3.3.4.

3.3.2. DASH Client Guidelines

The DASH-related aspects in DASH-IF IOPs as well as for the ISO BMFF based On-Demand and Live profiles of ISO/IEC 23009-1 are designed such that a client implementation can rely on relatively easy processes to provide an adaptive streaming service, namely:

- selection of the appropriate Adaptation Sets based on descriptors and other attributes
- initial selection of one Representation within each adaptation set
- download of (Sub)Segments at the appropriate time
- synchronization of different media components from different Adaptation Sets
- seamless switching of representations within one Adaptation Set

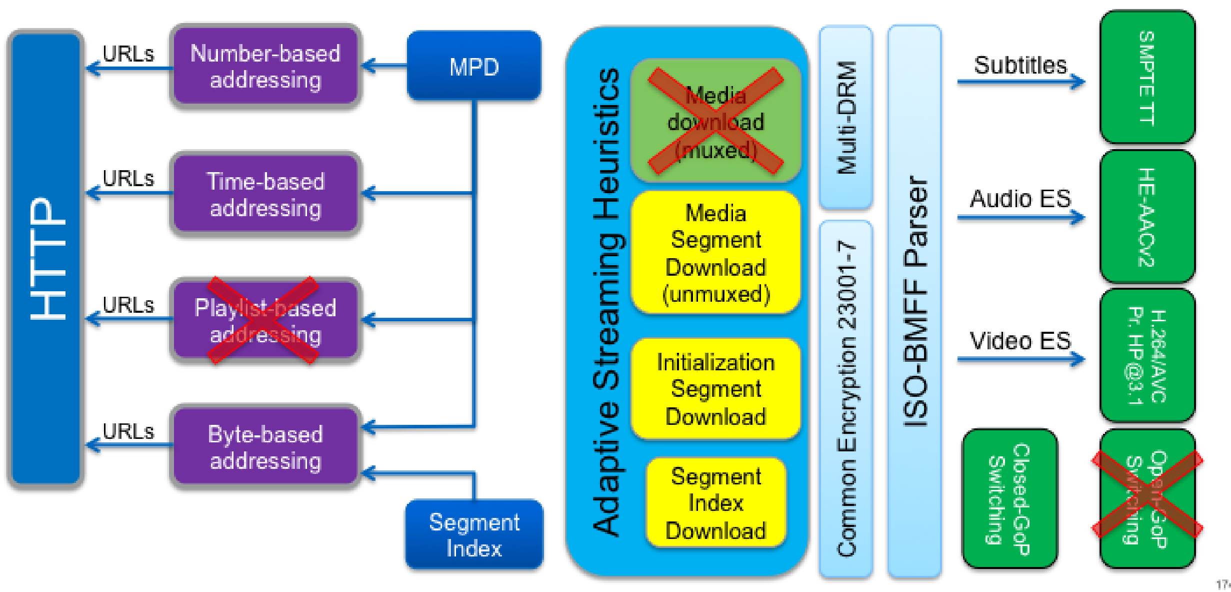

Figure 2 DASH aspects of a DASH-AVC/264 client compared to a client supporting the union of DASH ISO BMFF live and on-demand profile.

Figure 2 shows the DASH aspects of a DASH-AVC/264 client compared to a client supporting all features of the DASH ISO BMFF Live and On-Demand profile. The main supported features are:

- support of HTTP GET and partial GET requests to download Segments and Subsegments

- three different addressing schemes: number and time-based templating as well as byte range based requests.
- support of metadata as provided in the MPD and Segment Index
- download of Media Segments, Initialization Segments and Segment Index
- ISO BMFF parsing
- synchronized presentation of media components from different Adaptation Sets
- switching of video streams at closed GOP boundaries

3.3.3. Seamless switching

The formats defined in section 3.2 are designed for providing good user experience even in case the access bandwidth of the DASH Segment delivery or the cache varies. A key functionality is the ability that the DASH client can seamlessly switch across different Representations of the same media component. DASH clients should use the common timeline across different Representation representing the same media component to present one Representation up to a certain time t and continue presentation of another Representation from time *t* onwards. However, in practical implementations, this operation may be complex, as switching at time *t* may require parallel download and decoding of two Representations. Therefore, providing suitable switching opportunities in regular time intervals simplifies client implementations.

The formats defined in section 3.2 provide suitable switching opportunities at (sub)segment boundaries.

3.3.4. DASH Client Requirements

In order to ensure a minimum level of interoperability, a DASH-IF conforming client shall at least support the following features:

- The DASH client, if it switches, shall provide a seamless experience. A DASH shall be able to switch seamlessly at (sub)segment boundaries according to the definition in ISO/IEC 23009-1 [4], clause 4.5.1.
- If the scheme or the value for the following descriptor elements are not recognized and no equivalent other descriptor is present, the DASH client shall ignore the parent element:
    - **`FramePacking`**
    - **`Rating`**
    - **`EssentialDescriptor`**
    - **`ContentProtection`**

3.4. Transport and Protocol-Related Issues

3.4.1. General

Servers and clients operating in the context of the interoperability points defined in this document shall support the normative parts of HTTP/1.1 as defined in RFC 7230 [21], RFC 7231 [22], RFC 7232 [23], RFC 7233 [24], and RFC 7234 [25].

Specific requirements and recommendations are provided below.

> Note: IETF recently obsoleted RFC 2616 and replaced it with the six RFCs referred above. The changes are generally text clarifications and in some cases, additional constraints to address security or interoperability issues. Each new RFC contains details of the changes

compared to RFC2616. The IETF strongly recommends to reference and use the new RFCs that collectively replace RFC2616. This version of DASH-IF IOP addresses this aspect.

MPEG-DASH explicitly permits the use of https as a scheme and hence, HTTP over TLS as a transport protocol as defined in RFC 2818 [65]. For more details refer to section 7.2.

3.4.2. Server Requirements and Guidelines

HTTP Servers serving segments should support suitable responses to byte range requests (partial GETs).

If an MPD is offered that contains Representations conforming to the ISO BMFF On-Demand profile, then the HTTP servers offering these Representations shall support suitable responses to byte range requests (partial GETs).

HTTP Servers may also support the syntax using Annex E of 23009-1 using the syntax of the second example in Annex E.3,

```
BaseURL@byteRange="$base$?$query$&range=$first$-$last$"
```

3.4.3. Client Requirements and Guidelines

Clients shall support byte range requests, i.e. issue partial GETs to subsegments as defined in RFC 7233 [24]. Range requests may also be issued by using Annex E of 23009-1 using the syntax of the second example in Annex E.3,

```
BaseURL@byteRange="$base$?$query$&range=$first$-$last$"
```

Clients shall follow the reaction to HTTP status and error codes as defined in section A.7 of ISO/IEC 23009-1.

Clients should support the normative aspects of the HTTP state management mechanisms (also known as Cookies) as defined in RFC 6265 [41] for first-party cookies.

3.4.4. Transforming Proxies and Other Adaptation Middleboxes

A number of video transcoding proxies (aka "middleboxes") are already deployed on the wider Internet may silently transcode Representations. Specifically: a middlebox may see a video/mp4 response, transcode that video into a different format (perhaps using a lower bitrate or a different codec), then forward the transcoded video to the DASH client. This will break MPD and/or Segment Index based byte range operations, as those ranges are generally not valid in the transcoded video.

If such a threat is possible, one of the following approaches may be considered in order to prevent proxies from transcoding DASH Representations:

1. serve Media Presentations using encryption (e.g., HTTP over TLS, segment encryption or content protection),
2. serve Representations with Cache-Control: “`no-transform`"

In all cases the operational impacts on caching and implementations should be considered when using any of the above technologies.

In order to prevent middleboxes to manipulate the MPD, e.g. removing certain Representations or Adaptation Sets, the MPD may be securely transported by appropriate means, e.g. HTTPS.

3.5. Synchronization Considerations

In order to properly access MPDs and Segments that are available on DASH servers, DASH servers and clients should synchronize their clocks to a globally accurate time standard. Specifically it is expected that the Segment Availability Times as computed from the MPD according to ISO/IEC 23009-1 [4], section 5.3.9.5 and additional details in ISO/IEC 23009-3 [6], section 6.4 are accurately announced in the MPD.

Options to obtain timing for a DASH client are for example:

- Usage of NTP or SNTP as defined in RFC5905 [40].
- The Date general-header field in the HTTP header (see RFC 7231 [22] section 7.1.1.2) represents the date and time at which the message was originated, and may be used as an indication of the actual time.

Anticipated inaccuracy of the timing source should be taken into account when requesting segments close to their segment availability time boundaries.

More details on advanced synchronization support is provided in section 4.7.

3.6. Considerations for Live Services

For interoperability aspects of live services, please refer to section 4.

3.7. Considerations on Ad Insertion

For interoperability aspects for ad insertion use cases, please refer to section 5.

3.8. Switching across Adaptation Sets

Note: This technology is expected to be available in ISO/IEC 23009-1:2014/Amd.4:2016 [4], section 5.3.3.5. Once published by MPEG, this section is expected to be replaced by a reference to the MPEG-DASH standard.

Representations in two or more Adaptation Sets may provide the same content. In addition, the content may be time-aligned and may be offered such that seamless switching across Representations in different Adaptation Sets is possible. Typical examples are the offering of the same content with different codecs, for example H.264/AVC and H.265/HEVC and the content author wants to provide such information to the receiver in order to seamlessly switch Representations (as defined in ISO/IEC 23009-1, clause 4.5.1) across different Adaptation Sets. Such switching permission may be used by advanced clients.

A content author may signal such seamless switching property across Adaptation Sets by providing a Supplemental Descriptor along with an Adaptation Set with `@schemeIdUri` set to `urn:mpeg:dash:adaptation-set-switching:2016` and the `@value` is a comma-separated list of Adaptation Set IDs that may be seamlessly switched to from this Adaptation Set.

If the content author signals the ability of Adaptation Set switching and as `@segmentAlignment` or `@subsegmentAlignment` are set to `TRUE` for one Adaptation Set, the (Sub)Segment alignment shall hold for *all* Representations in *all* Adaptation Sets for which the `@id` value is included in the `@value` attribute of the Supplemental descriptor.

As an example, a content author may signal that seamless switching across an H.264/AVC Adaptation Set with **AdaptationSet**`@id="264"` and an HEVC Adaptation Set with **AdaptationSet**`@id="265"` is possible by adding a Supplemental Descriptor to the H.264/AVC Adaptation Set with `@schemeIdUri` set to `urn:mpeg:dash:adaptation-set-switching:2016` and the `@value="265"` and by adding a Supplemental Descriptor to the HEVC Adaptation Set with `@schemeIdUri` set to `urn:mpeg:dash:adaptation-set-switching:2016` and the `@value="264"`.

In addition, if the content author signals the ability of Adaptation Set switching for

- any Video Adaptation Set as defined in 3.2.13 then the parameters as defined in section 3.2.4 for an Adaption Set shall also hold for all Adaptation Sets that are included in the `@value` attribute.
- any Audio Adaptation Set as defined in 3.2.13 then the parameters as defined in section 3.2.4 for an Adaption Set shall also hold for all Adaptation Sets that are included in the `@value` attribute.

Note that this constraint may result that the switching may only be signaled with one Adaptation Set, but not with both as for example one Adaptation Set signaling may include all spatial resolutions of another one, whereas it is not the case the other way round.

3.9. Annotation and Client Model for Content Selection

3.9.1. Introduction

Beyond the ability to provide multiple Representations of the same media component in one Adaptation Set, DASH MPDs also provide the functionality to annotate Adaptation Sets, such that clients can typically select at most one Adaptation Set for each media type, based on the encoding and description provided in the MPD. The selection is based on client capabilities, client preferences, user preferences and possibly also interactive signalling with the user. Typically, the signalling and selection is independent of the codec in use. This clause provides requirements and recommendations for labelling Adaptation Sets, if multiple tracks are offered. Note that there may be cases that multiple Representations from different Adaptation Sets per media type are chosen for playback, for example if there is a dependency across Representations. In other cases, a DASH client may be asked to select more than one Adaptation Set per media type based on application decisions.

Multiple Adaptation Sets may be offered to provide the same content in different encodings, for example different codecs; or different source formats, for example one Adaptation Set encoded from a standard dynamic range master and another encoded from a high dynamic range video master. Alternatively, Adaptation Sets may describe different content, for example different languages, or different camera views of the same event that are provided in a synchronized presentation in one MPD.

Proper labelling of Adaptation Sets in MPDs conforming to DASH-IF IOPs is essential in order to enable consistent client implementations. In addition, also a model is needed on how the client makes use of the annotation for a content authors to understand the expected effect of the labelling on playback.

3.9.2. Adaptation Set Labeling Options for Selection

DASH in ISO/IEC 23009-1 [4] provides many options for labelling Adaptation Sets. In order to provide more consistency in the context of DASH-IF, Table 4 provides a restricted subset of labels for which DASH-IF IOPS provide interoperability, i.e. on how they are expected to be used by the content authors and how they are expected to be used by clients. The table provides information specific for each media type.

It is expected that DASH clients following the DASH IOPs recognize the descriptors, elements, and attributes as documented in Table 4.

Other organizations may define additional descriptors or elements, as well as processing models for clients.

Table 4 Adaptation Set Attributes and Elements and Usage in DASH-IF IOPs (see ISO/IEC 23009-1 [4])

Attribute or Element	Use for media type	Detailed Usage in DASH-IF IOPs
General Attributes and Elements for any media type		
`@profiles`	O	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. If not present, it is inherited from the MPD or Period. This may be used for example to signal extensions for new media profiles in the MPD. At least one of the values defined in Table 1 and Table 2 of this document shall be present, or inferred from MPD or Period higher-level.
`@group`	O	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. The attribute may be used and shall be different at least for different media type. If present, the value shall be greater than 0. For all Adaptation Sets in the same group, the Group shall be the same. Only one Representation in a Group is intended to be presented at a time. However, two or multiple groups of the same media type may exist, if the content author expects simultaneous presentation of two or more Representation of the same media type.
`@selectionPriority`	OD default=1	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. This attribute should be used to dis-ambiguate Adaptation Sets within one group for selection and expresses the preference of the MPD author on selecting Adaptation Sets for which the DASH client does make a decision otherwise. Examples include two video codecs providing the same content, but one of the two provides higher compression efficiency and is therefore preferred by the MPD author.
`ContentProtection`	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. If this element is present, then the content is protected. If not present, no content protection is applied. For details and usage please refer to clause 7.

Attribute or Element	Use for media type	Detailed Usage in DASH-IF IOPs
EssentialProperty	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9 specifies information about the containing element that is considered essential by the Media Presentation author for processing the containing element. The following schemes are expected to be recognized by a DASH-IF client independent of the media type: • http://dashif.org/guidelines/trick-mode (see clause 3.2.9)
SupplementalProperty		See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9 specifies information about the containing element that is considered supplemental by the Media Presentation author for processing the containing element. In no case this information is used for differentiation, the information may used by a DASH client for improved operation. The following schemes are expected to be recognized by a DASH-IF client independent of the media type: - urn:mpeg:dash:adaptation-set-switching:2016 (see clause 3.8) - http://dashif.org/guidelines/trick-mode (see clause 3.2.9) - urn:mpeg:dash:period-continuity:2015 (see clause 3.2.12) - urn:mpeg:dash:period-connectivity:2015 (see clause 3.2.12)
Viewpoint	0 … N	Provides the ability to indicate that media differentiates by a different ViewPoint. If not present, no view point is assigned and no differentiation is taken. For detailed usage of this descriptor, see below.
Label	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. This element enables to provide a textual description of the content. This element should be used if content author expects that clients supports UI for selection. However, this element must not be used as the sole differentiating element as at start-up no user interaction is available.
Attributes and Elements for media type "Video"		
@mimeType	M	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. Shall be set to "video/mp4".
@codecs	M	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9.

Attribute or Element	Use for media type	Detailed Usage in DASH-IF IOPs
		This provides the codec that is used for the Adaptation Set. It expresses the codec that is necessary to playback all Representations in one Adaption Set. The following codecs are expected to be recognized by a DASH-IF client: • Codecs in Table 17 • Codecs in Table 19
`@par`	O	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. shall be present, if the display aspect ratio is a differentiating parameter in the MPD.
`@maxWidth`	O	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. This attribute should be present to express the maximum width in samples after decoder sample cropping of any Representation contained in the Adaptation Set. The value should be the maximum horizontal sample count of any SPS in the contained bitstream.
`@maxHeight`	O	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. This attribute should be present to express the maximum height in pixel of any Representation contained in the Adaptation Set. The value should be the maximum horizontal sample count of any SPS in the contained bitstream.
`@maxFrameRate`	O	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. This attribute should be present to express the maximum frame rate, i.e. the maximum value of any entry in the Decoder configuration record of the signaled frame rate, if constant frame rate is provided. contained in the Adaptation Set.
`@scanType`	OD Default: progressive	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. This value is expected to be not present. If present, it is expected to be set to "`progressive`".
`EssentialProperty`	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. specifies information about the containing element that is considered essential by the Media Presentation author for processing the containing element. The following schemes are expected to be recognized by a DASH-IF client for video: • `urn:mpeg:mpegB:cicp:<Parameter>` as defined in ISO/IEC 23001-8 [49] and <Parameter> one of

Attribute or Element	Use for media type	Detailed Usage in DASH-IF IOPs
		the following: `ColourPrimaries`, `Transfer-Characteristics`, or `MatrixCoefficients`
`Accessibility`	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. In DASH-IF IOPs two schemes for accessibility are defined. - the Role scheme as defined by MPEG-DASH should be used as defined in ISO/IEC 23009-1, 5.8.5.5, `urn:mpeg:dash:role:2011` The DASH role scheme and with the following values is expected to be recognized by a DASH-IF client for media type "video" together with the Accessibility descriptor: o `sign` o `captions` - the scheme when CEA-608 is used as defined in clause 6.4.3.3, with `@schemeIdURI` set to `"urn:scte:dash:cc:cea-608:2015"`
`Role`	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. In DASH-IF IOPs only the Role scheme as defined by MPEG-DASH should be used as defined in ISO/IEC 23009-1 [4], 5.8.5.5, `urn:mpeg:dash:role:2011` The DASH role scheme and with the following values is expected to be recognized by a DASH-IF client for media type "video" together with the Role descriptor: - `caption` - `subtitle` - `main` - `alternate` - `supplementary` - `sign` - `emergency` If not present, the role is assumed to be `main`
`Rating`	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. DASH-IF IOPs do not define a Rating scheme. If present, Adaptation Sets using this descriptor may be ignored by the DASH-IF IOP clients.
Attributes and Elements for media type "Audio"		
`@mimeType`	M	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9.

Attribute or Element	Use for media type	Detailed Usage in DASH-IF IOPs
		Shall set to `"audio/mp4"`.
`@codecs`	M	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. This provides the codec that is used for the Adaptation Set. It expresses the codec that is necessary to playback all Representations in one Adaption Set. The following codecs are expected to be recognized by a DASH-IF client: • Codecs in Table 20 • Codecs in Table 23 • Codecs in Table 24 • Codecs in Table 25 • Codecs in Table 26 • Codecs in Table 27 Note: additional values need to be added with new codecs being added
`@lang`	O	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. The language should be present. If not present, the language is unknown or no language applies.
`@audioSamplingRate`	O	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. This attribute may be present to support output devices that may only be able to render specific values.
AudioChannelConfiguration	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. specifies information about the Audio channel configuration.The following schemes are expected to be recognized by a DASH-IF client for audio: • `urn:mpeg:dash:23003:3:audio_channel_configuration:2011` as defined in ISO/IEC 23009-1 [4], 5.8.5.4 • `urn:mpeg:mpegB:cicp:ChannelConfiguration` as defined in ISO/IEC 23001-8 [49] • `tag:dolby.com,2014:dash:audio_channel_configuration:2011` as defined at http://dashif.org/identifiers/audio-source-data/ Note: Annotation may be different for other codecs and may be updated
EssentialProperty	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9.

Attribute or Element	Use for media type	Detailed Usage in DASH-IF IOPs
		specifies information about the containing element that is considered essential by the Media Presentation author for processing the containing element. The following schemes are expected to be recognized by a DASH-IF client for audio: • `urn:mpeg:dash:audio-receiver-mix:2014` as defined in ISO/IEC 23009-1, clause 5.8.5.7.
`Accessibility`	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. In DASH-IF IOPs only the Role scheme as defined by MPEG-DASH should be used as defined in ISO/IEC 23009-1, 5.8.5.5, `urn:mpeg:dash:role:2011` The DASH role scheme and with the following values is expected to be recognized by a DASH-IF client for media type "audio" together with the Accessibility descriptor: • `description` • `enhanced-audio-intelligibility`
`Role`	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. In DASH-IF IOPs only the Role scheme as defined by MPEG-DASH should be used as defined in ISO/IEC 23009-1, 5.8.5.5, `urn:mpeg:dash:role:2011` The DASH role scheme and with the following values is expected to be recognized by a DASH-IF client for media type "audio" together with the Accessibility descriptor: - `main` - `alternate` - `supplementary` - `commentary` - `dub` - `emergency` If not present, the role is assumed to be `main`
`Rating`	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. DASH-IF IOPs do not define a Rating scheme. If present, Adaptation Sets using this descriptor may be ignored by the DASH-IF IOP clients.
Attributes and Elements for media type "Subtitle"		
`@mimeType`	M	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9.

Attribute or Element	Use for media type	Detailed Usage in DASH-IF IOPs
		Shall set to "application/mp4" or "application/ttml+xml"
@codecs	O	See ISO/IEC 23009-1 [4], clause 5.3.7.2 Table 9. This provides the codec that is used for the Adaptation Set. It expresses the codec that is necessary to playback all Representations in one Adaption Set. The following codecs are expected to be recognized by a DASH-IF client: • Codecs in Table 21 Note: more need to be added with new codecs being added.
@lang	O	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. The language should be present. If not present, the language is unknown or no language applies.
Accessibility	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. In DASH-IF IOPs only the Role scheme as defined by MPEG-DASH should be used as defined in ISO/IEC 23009-1, 5.8.5.5, urn:mpeg:dash:role:2011 The DASH role scheme and with the following values is expected to be recognized by a DASH-IF client for media type “subtitle” together with the Accessibility descriptor: - caption - sign
Role	0 … N	See ISO/IEC 23009-1 [4], clause 5.3.3.2 Table 5. In DASH-IF IOPs only the Role scheme as defined by MPEG-DASH should be used as defined in ISO/IEC 23009-1, 5.8.5.5, urn:mpeg:dash:role:2011 The DASH role scheme and with the following values is expected to be recognized by a DASH-IF client for media type “subtitle” together with the Accessibility descriptor: - main - alternate - subtitle - supplementary - commentary - dub - description

Attribute or Element	Use for media type	Detailed Usage in DASH-IF IOPs
		- `emergency` If not present, the role is assumed to be `main`

3.9.3. Content Model

In order to support the content author in providing content in a consistent manner, Figure 1 provides a conceptual content model for DASH content in one Period of an MPD. The content may be described by an Asset Identifier as a whole and may contain different media types, video, audio, subtitle and application types. Signalling of media types is out of scope for this section, for details refer to section 3.2.12.

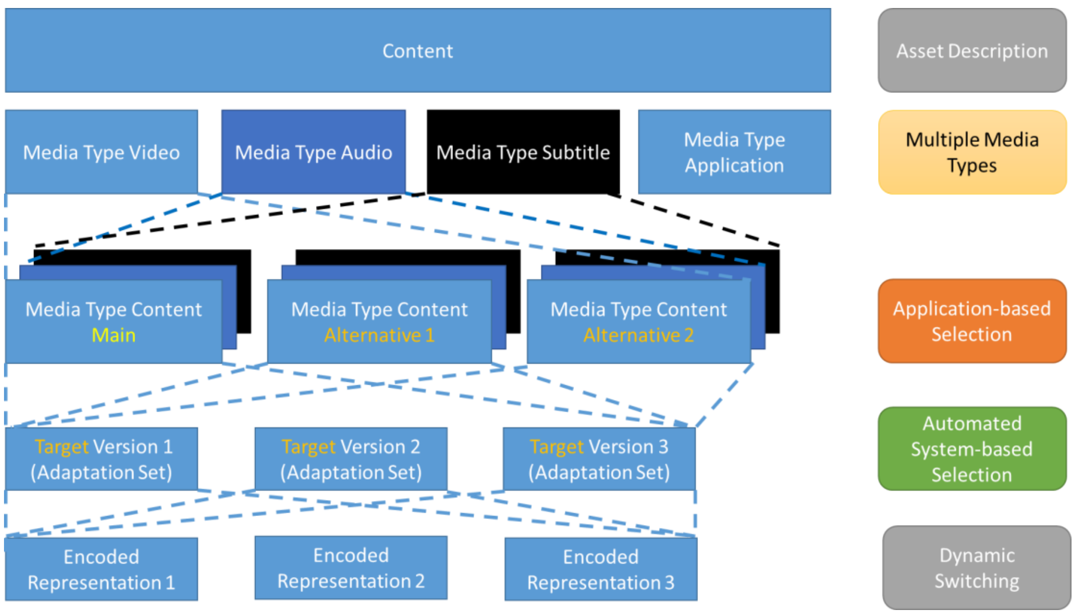

Figure 3 Content Model for DASH Multitrack

Within each media type, the content author may want to offer different alternative content that are time-aligned, but each alternative represents different content. Automatic selection of the alternative content is not expected to be done by the DASH client as the client would not have sufficient information to make such decisions. However, the selection is expected to be done by communication with an application or the user, typically using a user interface appropriate for selection.

However, in the absence of this external communication, or at startup, the DASH client still needs to playback content and therefore benefits from information of what is the default content. Such signalling should be provided by the content author. Such default content is referred to as main content, whereas any content that is not main is referred to as alternative. There may be multiple alternatives which may need to be distinguished. We define *main* and *alternative* content. Examples for such are synchronized camera views of one master content. The main camera view is provided as main content, all other views as alternative content.

Furthermore, it may be that content of different media type is linked by the content author, to express that two content of different media type are preferably played together. We define *associated* content for this purpose. As an example, there may be a main commentator associated with the main camera view, but for a different camera view, a different associated commentary is provided.

In addition to semantical content level differentiation, each alternative content may be prepared with different target versions, based on content preparation properties (downmix, subsampling, translation, suitable for trick mode, etc.), client preferences (decoding or rendering preferences, e.g. codec), client capabilities (DASH profile support, decoding capabilities, rendering capabilities) or user preferences (accessibility, language, etc.). In simple AV playout and in the absence of guidance from an application, a content author expects that the DASH client selects at most one target version for each Group taking into account its capabilities and preferences and the capabilities and preferences of the media subsystem. However, an application may obviously select multiple Groups and playout different video Adaptation Sets to support for example picture-in-picture, multi-angle and so on.

In addition, the content author may also provide priorities for target versions, if the receivers support multiple of those. Typical examples are that the content is prepared for H.264/AVC and H.265/HEVC capable receivers, and the content author prefers the selection of the H.265/HEVC version as its distribution is more efficient. A device supporting both decoders may then choose the one with higher priority signalled by the content author. In a similar version, the same content may be provided in different languages. In this case, it can still be expected that the language can be automatically selected by the client, so it is assigned to a target version. Again, a content author may express priorities on languages, for example preferring the native language over a dubbed one. Languages may be considered as alternative content as well, but as long as automatic selection can be provided, it may be considered as different target versions. Hence for each content of one media type, different *target versions* may exist and the annotation of the content expressed that it is expected that automated selection can be done. Each target version is preferably accumulated in one Adaptation Set, with exceptions such as scalable codecs.

Finally, in the content model, each of the target version typically has multiple Representations that are prepared to enable dynamic switching. This aspect is outside the scope of this section as switching by the client is expected to be done independent of the media type as well as the target version, primarily using the bandwidth and possibly abstract quality information. However, the signalling on the target versions may provide information on how to distribute the available bitrate across different media types.

Based on this content model and the available elements, attributes and descriptors from Table 4, requirements and recommendations are provided for Adaptation Set Signalling to address main and alternative content, associated content as well as different target versions. Based on the signalling, a client decision model is developed that may serve a content provider as a reference client to test if the annotation provided in the MPD provides the proper results.

3.9.4. Signalling Requirements and Recommendations

3.9.4.1. General

Assuming the content author can map its content to the above content model, this section provides signalling requirements and recommendations for such content, such that the content author can expect proper playback of its content for DASH-IF IOP clients

In general, if multiple Adaptation Sets for one media types are provided, sufficient information should be provided such that a DASH client make proper selections, either automatically communicating with its platform or in communication with the application/user.

3.9.4.2. Alternative Content Signalling

If a Period contains alternative content for one media type, then the alternatives should be differentiated. In addition, one of the alternatives should be provided as main content. The main content is intended to be selected by the client in the absence of any other information, e.g. at startup or if the annotation of the content cannot be used.

Main content is signaled by using the Role descriptor with Role scheme as defined by MPEG-DASH in ISO/IEC 23009-1, 5.8.5.5, `urn:mpeg:dash:role:2011` with value set to `main`. *Alternative content* is signaled by using the Role descriptor with Role scheme as defined by MPEG-DASH in ISO/IEC 23009-1, 5.8.5.5, `urn:mpeg:dash:role:2011` with value set to `alternative`. If an Adaptation Set does not include either of the two signals, it is assumed to be *main content*.

The alternative content may be selected by the client, if the client does have the capability to select alternatives, typically by either communicating with the application or with the user. If main and alternative content is provided in the Media Presentation, then alternative content shall be signaled by at least one of the two:

- a **`ViewPoint`** descriptor. If **`ViewPoint`** is used for differentiation, then at least each alternative Adaptation Set of the same media type shall include a **`ViewPoint`** with the same value for `@schemeIdURI`. The content is differentiated by different values for the `@value` attribute in the descriptor for different content.

- a **`Label`** element. If **`Label`** is used for differentiation, then at least each alternative Adaptation Set shall include a **`Label`** with the same value for `@id`. The content is differentiated by different values for the **`Label`** element.

A **`ViewPoint`** descriptor is typically used if a target application (identified by the the value for `@schemeIdURI`) is expected that can make use of the values in the ViewPoint descriptor. A **`Label`** element is typically used if the DASH client can provide a user interaction.

3.9.4.3. Associated Content Signalling

For associated content of different media types, the **`ViewPoint`** descriptor is used. If different media types all belong to one alternative content, they share the same View Point descriptor, i.e. the same value for `@schemeIdURI` and for `@value`. Note also that even if the DASH client does not understand the value for `@schemeIdURI` it would stll obey the rules for associated selection. The DASH client may for example use the labels of different video alternatives for selection, and play the audio according to ViewPoint association.

3.9.4.4. Media-type Independent Target Version Annotation

Adaptation Sets within one media type and alternative content shall differ by at least by one of the following annotation labels

- `@profiles`,
- **`ContentProtection`** (need to provide some details on what the options are: present, not-present, different schemes) ➔ work with content protection task force
- **`EssentialProperty`** (not-present, trickmode, a media type specific value, unknown value, which may be extended)
- Any of those documented in section 3.10.4.5 for media type video, section 3.10.4.6 for media type audio and 3.10.4.7 for media type subtitle.

Adaptation Sets with elements **`EssentialProperty`** not using any of the permitted values in this document should not be present.

In addition, Adaptation Sets within one media type and alternative content should differ by different values of `@selectionPriority`. If not present or non-differentiating values are provided, then the content author should expect a random selection of Adaptation Sets in case it is able to handle multiple Adaptation Sets within one media type and alternative content.

3.9.4.5. Video Target Version Annotation

Video Adaptation Sets of one alternative content shall differ by at least by one of the following annotation labels:

- `@codecs`: specifies the codecs present within the Representation. The codecs parameters shall also include the profile and level information where applicable.
- `@maxWidth` and `@maxHeight` specifies the horizontal and vertical visual presentation size of the video media type
- `@maxFrameRate` specifies the maximum frame rate of the video media type
- **`EssentialProperty`**: specifies information about the containing element that is considered essential by the Media Presentation author selecting this component.
    - The following different options exist: not-present; generic parameters from above; list in Table 1; unknown value, which may be extended
- Accessibility descriptor with
    - Role scheme as defined by MPEG-DASH in ISO/IEC 23009-1, 5.8.5.5, `urn:mpeg:dash:role:2011` with value set to `sign`, `caption` or `subtitle`. The presence of `caption` or `subtitle` signals open ("burned in") captions or subtitles.

- the scheme when CEA-608 is used as defined in clause 6.4.3.3, with `@schemeIdURI` set to `"urn:scte:dash:cc:cea-608:2015"` indicating the use of CEA-608 captions carried in SEI messages.

Adaptation Sets with elements **`Rating`** and **`FramePacking`** as well with `@scanType` not set to `progressive` should not be present.

The content author should use the `@selectionPriority` attribute in order to express preference for video selection. If captions are burned in with video Adaptation Set, and other video Adaptation Sets are available as well, the content author should use the `@selectionPriority` to indicate the selection priority of this Adaptation Set compared to others without burned in captions.

3.9.4.6. Audio Target Version Annotation

Audio Adaptation Sets of one alternative content shall differ by at least by one of the following annotation labels:

- `@codecs`: specifies the codecs present within the Representation. The codecs parameters shall also include the profile and level information where applicable.
- `@lang`: specifies the dominant language of the audio
  - If not present, the language is unknown or no language applies
- `@audioSamplingRate` specifies the maximum sampling rate of the content
  - If not present, the audio sampling rate is unknown
- The **`AudioChannelConfiguration`** specifies support for output devices that may only be able to render specific values. This element should be present.
  - If no **`AudioChannelConfiguration`** is present, then this value is unknown.
  - If the codec is anyone in Table 20, Table 25, Table 26 or Table 27, then any of the following may be used
    - `urn:mpeg:dash:23003:3:audio_channel_configuration:2011` as defined in ISO/IEC 23009-1 [1], 5.8.5.4
    - `urn:mpeg:mpegB:cicp:ChannelConfiguration` as defined in ISO/IEC 23001-8 [49]
  - If the codec is ec-3 or ac-4 according to Table 23, then the following shall be used
    - `tag:dolby.com,2014:dash:audio_channel_configuration:2011`" as defined at http://dashif.org/identifiers/audio-source-data/ (see section 9.2.1.2)
  - If the codec is anyone in Table 24, then refer to DTS specification 9302K62400 [39]

- **`EssentialProperty`**: specifies information about the containing element that is considered essential by the Media Presentation author selecting this component.
    - The following different options exist: not-present; generic parameters from above; unknown value, which may be extended
- Accessibility descriptor with Role scheme as defined by MPEG-DASH in ISO/IEC 23009-1, 5.8.5.5, `urn:mpeg:dash:role:2011` with value set to `description` or `enhanced-audio-intelligibility`.

Note that Adaptation Sets with element **`Rating`** may be ignored by the client and should therefore only be used if the content provider has knowledge that clients can process the applied Rating scheme.

3.9.4.7. Subtitle Target Version Annotation

Subtitle Adaptation Sets of one alternative content shall differ by at least by one of the following annotation labels:

- `@codecs`: specifies the codecs present within the Representation. The codecs parameters shall also include the profile and level information where applicable.
- `@lang`: specifies the language of the subtitle
    - If not present, the language is unknown or no language applies
- **`EssentialProperty`**: specifies information about the containing element that is considered essential by the Media Presentation author selecting this component.
    - The following different options exist: not-present; generic parameters from above; unknown value, which may be extended
- Accessibility descriptor with Role scheme as defined by MPEG-DASH in ISO/IEC 23009-1, 5.8.5.5, `urn:mpeg:dash:role:2011` with value set to `description` or `caption`.

3.9.4.8. Other Annotation, Auxiliary Data

In addition to selection relevant data, the Adaptation Set may also signal additional auxiliary information. Auxiliary information is expressed by

- The Role descriptor with the Role scheme as defined by MPEG-DASH as defined in ISO/IEC 23009-1, 5.8.5.5, `urn:mpeg:dash:role:2011` with the following values:
    - `caption`
    - `subtitle`
    - `main`
    - `alternate`
    - `supplementary`

- `sign`
- `emergency`
- `dub`

- The Supplemental descriptor with the `@schemeIdURI` and `@value` pairs:
  - Trickmode: `@schemeIdURI` set to "`http://dashif.org/guide-lines/trickmode`" and the `@value` the value of `@id` attribute of the Adaptation Set to which these trick mode Representations belong.
  - Period-continuous Adaptation Sets by using Aa `@schemeIdUri` set to "`urn:mpeg:dash:period-continuity:2015`" with the `@value` of the descriptor matching the value of an `@id` of a Adaptation Set that is contained in the MPD,
  - Period-connected Adaptation Sets by using Aa `@schemeIdUri` set to "`urn:mpeg:dash:period-connectivity:2015`" with the `@value` of the descriptor matching the value of an `@id` of a Adaptation Set that is contained in the MPD,
  - Switching across Adaptation Sets: `@schemeIdUri` set to `urn:mpeg:dash:adaptation-set-switching:2016` and the `@value` is a comma-separated list of Adaptation Set IDs that may be seamlessly switched to from this Adaptation Set.

3.9.5. Client Processing Reference Model

3.9.5.1. Introduction

The following client model serves two purposes:

- In the absence of other information, the following client model may be implemented in a DASH client for the purpose of selection of Adaptation Set for playout
- A content author may use the model to verify that the annotation is properly done in order to get the desired client behaviour.

In the model it is assumed that the client can get sufficient information on at least the following properties:

- For each codec in the `@codecs` string, the DASH client can get information if the media playback platform can decode the codec as described in the string. The answer should be yes or no.
- For each DRM system in the **`ContentProtection`** element string, the DASH client can get information if the media playback platform can handle this Content Protection scheme as described in the string. The answer should be yes or no.
- the DASH client can get information on the media playback platform and rendering capabilities in terms of

- the maximum spatial resolution for video that can be handled
  - the maximum frame rate for video that can be handled
  - the audio channel configuration of the audio system
  - the audio sampling rate of the audio system
- the preferred language of the system
- Accessibility settings for captions, subtitles, audio description, enhanced audio intelligibility,
- Potentially preferences on media playback and rendering of the platform

Note of any of these functionalities are not fulfilled, then it may still be functional, but it may not result in the full experience as provided by the content author. As an example, if the DASH client cannot determine the preferred language, it may just use the selection priority for language selection.

3.9.5.2. Generic Processing Model

The DASH client uses the MPD and finds the Period that it likes to join, typically the first one for On-Demand content and the one at the live edge for live content. In order to select the media to be played, the DASH client assumes that the content is offered according to the content model above.

1. The DASH client looks for *main content*, i.e. any Adaptation Set with annotation **Role**`@schemeIdURI="urn:mpeg:dash:role:2011"` and **Role**`@value="alternative"` is excluded initially for selection. Note that in this model it is assumed that immediate startup is desired. If the DASH client wants to go over the alternatives upfront before starting the service, then the sequence is slightly different, but still follows the remaining principles.
2. DASH Client checks each Adaptation Set for the supported capabilities of the platform
   - Codec support
   - DRM support
   - Rendering capabilities

   If any of the capabilities are not supported, then the Adaptation Set is excluded from the selection process.
3. The DASH client checks if it supports for CEA-608 rendering as defined in clause 6.4.3.3. If not supported, any accessibility descriptor with is `@schemeIdURI="urn:scte:dash:cc:cea-608:2015"` removed. Note that the Adaptation Set is maintained as it may used for regular video decoding.
4. DASH Client checks is there are any specific settings for accessibility in the user preferences
   - If captions are requested by the system, the DASH client extracts

- all video Adaptation Sets that have an **`Accessibility`** descriptor assigned with either the `@schemeIdURI=" urn:mpeg:dash:role:2011` and `@value="caption"` or `@schemeIdURI=" urn:scte:dash:cc:cea-608:2015"` (burned-in captions and SEI-based), as well as

    - all subtitle Adaptation Sets that have an **`Accessibility`** descriptor assigned with either the `@schemeIdURI=" urn:mpeg:dash:role:2011` and `@value="caption"`

    - and makes those available for Adaptation Sets that can be selected by the DASH client for caption support.

- If multiple caption Adaptation Sets remain, the DASH client removes all Adaptation Sets from the selection that are not in the preferred language, if language settings are provided in the system. If no language settings in the system are provided, or none of the Adaptation Sets meets the preferred languages, none of the Adaptation Sets are removed from the selection. Any Adaptation Sets that do not contain language annotation are removed, if any of the remaining Adaptation Sets provides proper language settings.

- If still multiple caption Adaptation Sets remain, then the ones with the highest value of `@selectionPriority` is chosen.

- If still multiple caption Adaptation Sets remain, then the DASH client makes a random choice on which caption to enable.

- else if no captions are requested

    - the **`Accessibility`** element signaling captions may be removed from the Adaptation Set before continuing the selection.

- If sign language is requested

    - all video Adaptation Sets that have an **`Accessibility`** descriptor assigned with `@schemeIdURI="urn:mpeg:dash:role:2011"` and `@value="sign"` are made available for sign language support.

- else if no sign language is requested

    - the Adaptation Set signaling sign language with the **`Accessibility`** element may be removed from the Adaptation Set before continuing the selection.

- If audio descriptions are requested

    - all video Adaptation Sets that have an **`Accessibility`** descriptor assigned with `@schemeIdURI="urn:mpeg:dash:role:2011"` and

`@value="description"` are made available for audio description support.

- else if no audio descriptions are requested
  - the Adaptation Set signaling audio descriptions with the **`Accessibility`** element may be removed from the Adaptation Set before continuing the selection.
- If enhanced audio intelligibility is requested
  - all audio Adaptation Sets that have an **`Accessibility`** descriptor assigned with `@schemeIdURI="urn:mpeg:dash:role:2011"` and `@value="enhanced-audio-intelligibility"` are made available for enhanced audio intelligibility support.
- else if no enhanced audio intelligibility is requested
  - the **`Accessibility`** element may be removed from the Adaptation Set before continuing the selection.

5. If video rendering is enabled, based on the remaining video Adaptation Sets the client selects one as follows:
   - Any Adaptation Set for which an Essential Descriptor is present for which the scheme or value is not understood by the DASH client, is excluded from the selection
   - Any Adaptation Set for which an Essential Descriptor is present for which the scheme is `http://dashif.org/guidelines/trickmode`, is excluded from the initial selection
   - If still multiple video Adaptation Sets remain, then the ones with the highest value of `@selectionPriority` is chosen.
   - If still multiple video Adaptation Sets remain, then the DASH client makes a choice for itself, possibly on a random basis.
   - Note that an Adaptation Set selection may include multiple Adaptation Sets, if Adaptation Set Switching is signaled. However, the selection is done for only one Adaptation Set.
6. If audio rendering is enabled, based on the remaining audio Adaptation Sets the client selects one as follows:
   - Any Adaptation Set for which an Essential Descriptor is present for which the scheme or value is not understood by the DASH client, is excluded from the selection

- If multiple audio Adaptation Sets remain, the DASH client removes all Adaptation Sets from the selection that are not in the preferred language, if language settings are provided in the system. If no language settings in the system are provided, or none of the Adaptation Sets meets the preferred languages, none of the Adaptation Sets are removed from the selection. Any Adaptation Set that does not contain language annotation are removed, if any of the remaining Adaptation Sets provides proper language settings.
- If still multiple audio Adaptation Sets remain, then the ones with the highest value of `@selectionPriority` is chosen.
- If still multiple audio Adaptation Sets remain, then the DASH client makes a choice for itself, possibly on a random basis.
- Note that an Adaptation Set may include multiple Adaptation Sets, if Adaptation Set Switching or receiver mix is signaled. However, the selection is done for only one Adaptation Set.

7. If subtitle rendering is enabled, based on the subtitle Adaptation Sets the client selects one as follows:
    - Any Adaptation Set for which an Essential Descriptor is present for which the scheme or value is not understood by the DASH client, is excluded from the selection
    - If multiple subtitle Adaptation Sets remain, the DASH client removes all Adaptation Sets from the selection that are not in the preferred language, if language settings are provided in the system. If no language settings in the system are provided, or none of the Adaptation Sets meets the preferred languages, none of the Adaptation Sets are removed from the selection. Any Adaptation Set that does not contain language annotation are removed, if any of the remaining Adaptation Sets provides proper language settings.
    - If still multiple subtitle Adaptation Sets remain, then the ones with the highest value of `@selectionPriority` is chosen.
    - If still multiple subtitle Adaptation Sets remain, then the DASH client makes a choice for itself, possibly on a random basis.
8. If the DASH client has the ability to possibly switch to alternative content, then alternative content may be selected either through the **`Label`** function or the **`ViewPoint`** functionality. This selection may be done dynamically during playout and the DASH client is expected to switch to the alternative content. Once all alternative content is selected, the procedures following from step 2 onwards apply.
9. At Period boundary a DASH client initially looks for a Period continuity or connectivity, i.e. does the Period include an Adaptation Set that is a continuation of the existing one. If not present it will go back to step 1 and execute the decision logic.

4. Live Services

4.1. Introduction

MPEG-DASH [1] provides several tools to support live services. This section primarily provides requirements and recommendations for both, content authoring as well as client implementations.

For this purpose, this section

- clarifies and refines details of interoperability points when used with the features available in the 2012 edition of MPEG-DASH with respect to different service configurations and client implementations.
- defines one new interoperability point in order to address content authoring and client requirements to support a broad set of live services based on the features defined in the second edition (published 2014) of MPEG-DASH as well certain amendments thereof.

The main features and differences of these two modes are provided in the following Table 5:

Table 5 Main features and differences of simple and main live services

Feature	Simple	Main
Support of `MPD@type`	static, dynamic	static, dynamic
MPD updates	yes	yes
MPD updated triggered	by MPD attribute minimum update period	by Inband Event messages in the segments.
URL generation	based on MPD	based on MPD and segment information
Timeline gaps	based on MPD and for entire content	may be signalled individually for each Representation
Segments starts with	closed GOP	closed GOP
Support of Simple Live	Yes	No
Support of Main Live	Yes	Yes

To support the definition of the interoperability points, architectures and use cases were collected. These are documented in Annex B.

4.2. Overview Dynamic and Live Media Presentations

DASH Media Presentations with `MPD@type` set to `"dynamic"` enable that media is made available over time and its availability may also be removed over time. This has two major effects, namely

1. The content creator can announce a DASH Media Presentation for which not all content is yet available, but only gets available over time.
2. Clients are forced into a timed schedule for the playout, such that they follow the schedule as desired by the content author.

Dynamic services may be used for different types of services:

1. **Dynamic Distribution of Available Content**: Services, for which content is made available as dynamic content, but the content is entirely generated prior to distribution. In this case the details of the Media Presentation, especially the Segments (duration, URLs) are known and can be announced in a single MPD without MPD updates. This addresses use cases 2 and 3 in Annex B.

2. **MPD-controlled Live Service**: Services for which the content is typically generated on the fly, and the MPD needs to be updated occasionally to reflect changes in the service offerings. For such a service, the DASH client operates solely on information in the MPD. This addresses the use cases 4 and 5 in Annex B.

3. **MPD and Segment-controlled Live**: Services for which the content is typically generated on the fly, and the MPD may need to be updated on short notice to reflect changes in the service offerings. For such a service, the DASH client operates on information in the MPD and is expected to parse segments to extract relevant information for proper operation. This addresses the use cases 4 and 5, but also takes into account the advanced use cases.

Dynamic and Live services are typically controlled by different client transactions and server-side signaling.

For initial access to the service and joining the service, an MPD is required. MPDs may be accessed at join time or may have been provided earlier, for example along with an Electronic Service Guide. The initial MPD or join MPD is accessed and processed by the client and the client having an accurate clock that is synchronized with the server can analyze the MPD and extract suitable information in order to initiate the service. This includes, but is not limited to:

- identifying the currently active Periods in the service and the Period that expresses the live edge (for more details see below)
- selecting the suitable media components by selecting one or multiple Adaptation Sets. Within each Adaptation Set selecting an appropriate Representation and identifying the live edge segment in each Representations. The client then issues requests for the Segments.

The MPD may be updated on the server based on certain rules and clients consuming the service are expected to update MPDs based on certain triggers. The triggers may be provided by the MPD itself or by information included in Segments. Depending on the service offering, different client operations are required as shown in Figure 4.

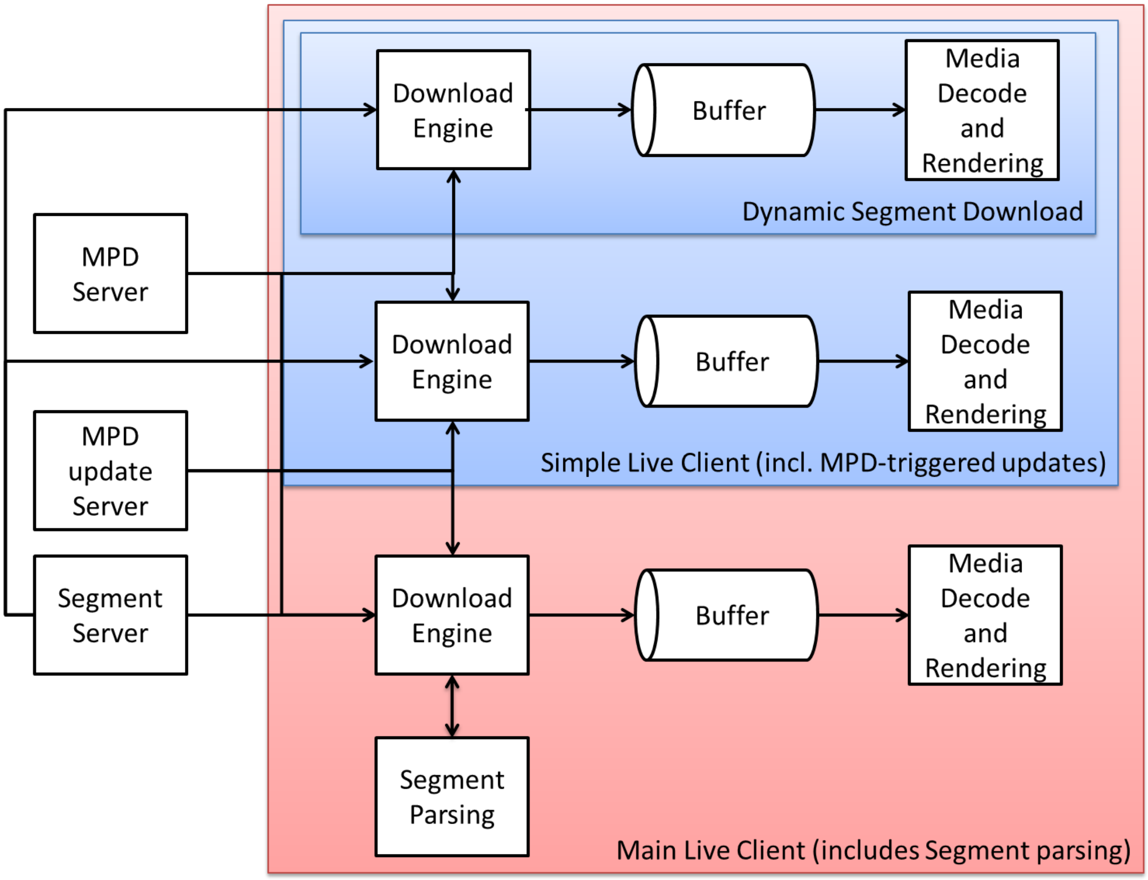

Figure 4 Different Client Models

The basic functions of a live client described in this document are as follows:

1. **Dynamic Segment Download:** This function creates a list of available Segments based on a single MPD and joins the service by downloading Segments at the live edge or may use the Segments that are available in the time shift buffer.
2. **Simple Live Client**: This client includes the dynamic segment download function and enables updates of the MPD based on information in the MPD in order to extend the Segment list at the live edge. MPDs are refetched and revalidated when the currently available MPD expires, i.e. an expired MPD can no longer be used for Segment URL generation.
3. **Main Live Client**: This client includes all features of the simple Live DASH client. In addition it generates Segment URLs and it updates the MPD based on information in the

Segments if the service offering provides this feature. MPDs are refetched and revalidated when the currently available MPD expires based on expiry information in the Segments.

Requirements and recommendations for the dynamic segment download functions are defined in in section 4.3.

Requirements and recommendations for simple live service offerings and corresponding clients are defined in section 4.4.

Requirements and recommendations for main live service offerings and corresponding clients are defined in section 4.5.

Requirements and recommendations when offering live services as on-demand are provided in section 4.6.

Requirements and recommendations for client-server timing synchronization are defined in section 4.7.

Requirements and recommendations for robust service offerings and corresponding clients are defined in section 4.8.

Interoperability Aspects are defined in section 4.9.

4.3. Dynamic Segment Download

4.3.1. Background and Assumptions

The dynamic segment download function is a key component of live services, In addition, the dynamic segment download function may also be used for scheduling a playout. In the remainder of this subsection, it is assumed that the client has access to a single instance of an MPD and all information of the entire Media Presentation is contained in the MPD.

We refer to this service as dynamic service as the main feature is that the Segments are made available over time following the schedule of the media timeline.

Dynamic services are primarily documented in order to provide insight into the timing model of Segment availabilities. This forms the basis for live services and explains the key concepts and rules for Segment availabilities.

4.3.2. Preliminaries

4.3.2.1. MPD Information

If the Media Presentation is of type dynamic, then Segments have different Segment availability times, i.e. the earliest time for which the service provider permits the DASH client to issue a request to the Segment and guarantees, under regular operation modes, that the client gets a 200 OK response for the Segment. The Segment availability times for each Representation can be computed based on the information in an MPD.

For a dynamic service the MPD should at least contain information as available in Table 6. Information included there may be used to compute a list of announced Segments, Segment Availability Times and URLs.

Assume that an MPD is available to the DASH client at a specific wall-clock time *NOW*. It is assumed that the client and the DASH server providing the Segments are synchronized to wall-clock, either through external means or through a specific client-server synchronization. Details on synchronization are discussed in section 4.7.

Assuming synchronization, the information in the MPD can then be used by the client at time *NOW* to derive the availability (or non-availability) of Segments on the server.

Table 6 -- Information related to Segment Information and Availability Times for a dynamic service

MPD Information	Status	Comment
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* in the following.
MPD@mediaPresentationDuration	mandatory (for the considered use cases)	provides the duration of the Media Presentation.
MPD@suggestedPresentationDelay	optional, but recommended	suggested presentation delay as delta to segment availability start time. The value is denoted as *SPD*. Details on the setting and usage of the parameter is provided in the following.
MPD@minBufferTime	mandatory	minimum buffer time, used in conjunction with the @bandwidth attribute of each Representation. The value is denoted as *MBT*. Details on the setting and usage of the parameter is provided in the following.
MPD@timeShiftBufferDepth	optional, but recommended	time shift buffer depth of the media presentation. The value is denoted as *TSB*. Details on the setting and usage of the parameter is provided in the following.
Period@start	Mandatory for the first Period in the MPD	the start time of the Period relative to the MPD availability start time.
Representation@availabilityTimeOffset	Optional default	The offset in availability time for this Representation. It may also be available on a Base URL or default. For

		more details refer to section 4.3.2.2.5. NOTE: the value of "`INF`" implies availability of all segments starts at **`MPD`**`@availabilityStartTime`
`SegmentTemplate``@media`	mandatory	The template for the Media Segment assigned to a Representation.
`SegmentTemplate``@startNumber`	optional default	number of the first segment in the Period assigned to a Representation
`SegmentTemplate``@timescale`	optional default	timescale for this Representation.
`SegmentTemplate``@duration` **`SegmentTemplate.SegmentTimeline`**	exactly one of **`SegmentTemplate`**`@duration` or **`SegmentTemplate.SegmentTimeline`** must be present per Representation.	the duration of each Segment in units of a time.

4.3.2.2. Segment Information Derivation

4.3.2.2.1. Introduction

Based on an MPD including information as documented in Table 6 and available at time *NOW* on the server, a synchronized DASH client derives the information of the list of Segments for each Representation in each Period. This section only describes the information that is expressed by the values in the MPD. The generation of the information on the server and the usage of the information in the client is discussed in section 4.3.3 and 4.3.4, respectively.

MPD information is provided in subsection 4.3.2.2.3. The Period based information is documented in sub-section 4.3.2.2.4, and the Representation information is documented in sub-section 4.3.2.2.5.

4.3.2.2.2. Definitions

The following definitions are relevant and aligned with ISO/IEC 23009-1:

- available Segment is a Segment that is accessible at its assigned HTTP-URL. This means that a request with an HTTP GET to the URL of the Segment results in a reply of the Segment and 2xx status code.
- valid Segment URL is an HTTP-URL that is promised to reference a Segment during its Segment availability period.
- *NOW* is a time that is expressing the time on the content server as wall-clock time. All information in the MPD related to wall-clock is expressed as a reference to the time *NOW*.

4.3.2.2.3. MPD Information

For a dynamic service without MPD updates, the following information shall be present and not present in the MPD (also please refer to Table 6):

- The **`MPD`**`@type` shall be set to "`dynamic`".

- The **MPD**@mediaPresentationDuration shall be present, or the **Period**@duration of the last Period shall be present.
- The **MPD**@minimumUpdatePeriod shall not be present.

Furthermore, it is recommended to provide a value for **MPD**@timeShiftBufferDepth and **MPD**@suggestedPresentationDelay.

4.3.2.2.4. Period Information

Each Period is documented by a **Period** element in the MPD. An MPD may contain one or more Periods. In order to document the use of multiple Periods, the sequence of Period elements is expressed by an index *i* with *i* increasing by 1 for each new Period element.

Each regular Period *i* in the MPD is assigned a

- Period start time *PSwc*[*i*] in wall-clock time,
- Period end time *PEwc*[*i*], in wall-clock time.

Note: An MPD update may extend the Period end time of the last regular Period. For details refer to section 4.4.

The Period start time *PSwc*[*i*] for a regular Period *i* is determined according to section 5.3.2.1 of ISO/IEC 23009-1:

- If the attribute @start is present in the **Period**, then *PSwc*[*i*] is the sum of *AST* and the value of this attribute.
- If the @start attribute is absent, but the previous **Period** element contains a @duration attribute then the start time of the Period is the sum of the start time of the previous Period *PSwc*[*i*] and the value of the attribute @duration of the previous Period. Note that if both are present, then the @start of the new Period takes precedence over the information derived from the @duration attribute.

The Period end time *PEwc*[*i*] for a regular Period *i* is determined as follows:

- If the Period is the last one in the MPD, the time *PEwc*[*i*] is obtained as
  - the sum of *AST* and Media Presentation Duration *MPDur*, with *MPDur* the value of **MPD**@mediaPresentationDuration if present, or the sum of *PSwc*[*i*] of the last Period and the value of **Period**@duration of the last Period.
- else
  - the time *PEwc*[*i*] is obtained as the Period start time of the next Period, i.e. *PEwc*[*i*] = *PSwc*[*i*+1].

4.3.2.2.5. Representation Information

Based on such an MPD at a specific time *NOW*, a list of Segments contained in a Representation in a Period *i* with Period start time *PSwc*[*i*] and Period end time *PEwc*[*i*] can be computed.

If the **SegmentTemplate.SegmentTimeline** is present and the **SegmentTemplate**@duration is not present, the **SegmentTimeline** element contains N_S **S** elements indexed with s=1, ..., N_S, then let

- *ts* the value of the @timescale attribute

- *ato* is the value of the `@availabilityTimeOffset` attribute, if present. Otherwise it is zero.
- *t*[*s*] be the value of `@t` of the *s*-th **`S`** element,
- *d*[*s*] be the value of `@d` of the *s*-th **`S`** element
- *r*[*s*] be,
  - if the `@r` value is greater than or equal to zero
    - one more than the value of `@r` of the *s*-th **`S`** element. Note that if `@r` is smaller than the end of this segment timeline element, then this Representation contains gaps and no media is present for this gap.
  - else
    - if *t*[*s*+1] is present, then *r*[*s*] is the ceil of (*t*[*s*+1] - *t*[*s*])/*d*[*s*]
    - else *r*[*s*] is the ceil of (*PEwc*[*i*] - *PSwc*[*i*] - *t*[*s*]/*ts*)**ts*/*d*[*s*])

If the **`SegmentTemplate`**`@duration` is present and the **`SegmentTemplate.SegmentTimeline`** is not present, then

- N_S=1,
- *ato* is the value of the `@availabilityTimeOffset` attribute, if present. Otherwise it is zero.
- *ts* the value of the `@timescale` attribute
- *t*[*s*] is 0,
- the *d*[*s*] is the value of `@duration` attribute
- *r*[*s*] is the ceil of (*PEwc*[*i*] - *PSwc*[*i*] - *t*[*s*]/*ts*)**ts*/*d*[*s*])

Note that the last segment may not exist and `r[s]` is one less than this computation provides. For more details, refer to clause 4.4.3.6.

4.3.2.2.6. Media Time Information of Segment

Each Media Segment at position *k=1,2, ...* for each Representation has assigned an earliest media presentation time *EPT*[*k*,*r*,*i*] and an accurate segment duration *SDUR[k,r,j]*, all measured in media presentation time.

The earliest presentation time may be estimated from the MPD using the segment availability start time minus the segment duration announced in the MPD.

The earliest presentation time may be accurately determined from the Segment itself.

For details on the derivation of the earliest presentation time, see section 3.2.11.

4.3.2.2.7. Segment List Parameters

For each Period *i* with Period start time *PSwc*[*i*] and Period end time *PEwc*[*i*] and each Representation *r* in the Period the following information can be computed:

- the presentation time offset described in the MPD, o[*i*,*r*]
- the availability time offset of this Representation, ato[*r*]
- the number of the first segment described in the MPD, k1[*i*,*r*]
- the number of the last segment described in the MPD, k2[*i*,*r*]

- segment availability start time of the initialization segment SAST[*0*,*i*,*r*]
- segment availability end time of the initialization segment SAET[*0*,*i*,*r*]
- segment availability start time of each media segment SAST[*k*,*i*,*r*], *k*=k1, ..., k2
- segment availability end time of each media segment SAET[*k*,*i*,*r*], *k*=k1, ..., k2
- adjusted segment availability start time ASAST[*0*,*i*,*r*], *k*=0, k1, ..., k2
- segment duration of each media segment SD[*k*,*i*,*r*], k=k1, ..., k2
- the URL of each of the segments, URL[*k*,*i*,*r*]

In addition,

- the latest available Period *i*[*NOW*] and the latest segment available at the server k[*NOW*] can be computed. This segment is also referred to as *live edge segment*.
- the earliest available Period *i**[*NOW*] and the earliest segment available at the server k*[*NOW*] can be computed.

Based on the above information, for each Representation *r* in a Period *i*, the segment availability start time *SAST*[*k*,*i*,*r*], the segment availability end time of each segment *SAET*[*k*,*i*,*r*], the segment duration of each segment *SD*[*k*,*i*,*r*], and the URL of each of the segments, *URL*[*k*,*i*,*r*] within one Period *i* be derived as follows using the URL Template function `URLTemplate`(`ReplacementString`, Address) as documented in subsection 4.3.2.2.8:

- *k*=0
- *SAST*[*0*,*i*,*r*] = *PSwc*[*i*]
- *ASAST*[*0*,*i*,*r*] = *PSwc*[*i*] - *ato*
- for *s=1*, ... N_s [*i*,*r*]
    - *k* = *k* + 1
    - *SAST*[*k*,*i*,*r*] = *PSwc*[*i*] + (*t*[*s*,*i*,*r*] + *d*[*s*,*i*,*r*] - o[*i*,*r*])/*ts*
    - *ASAST*[*k*,*i*,*r*] = *SAST*[*k*,*i*,*r*] – *ato*
    - *SD*[*k*,*i*,*r*] = *d*[*s*,*i*,*r*]/*ts*
    - *SAET*[*k*,*i*,*r*] = SAST[*k*,*i*,*r*] + *TSB* + *d*[*s*,*i*,*r*]/*ts*
    - if **`SegmentTemplate`**`@media` contains `$Number$`
        - *Address*=`@startNumber`
        - *URL*[*k*,*i*,*r*] = `URLTemplate` (`$Number$`, *Address*)

      else

        - *Address* = *t*[*s*,*i*,*r*]
        - *URL*[*k*,*i*,*r*] = `URLTemplate` (`$Time$`, *Address*)
    - for *j* = 1, ..., *r*[*s*,*i*,*r*]
        - *k* = *k* + 1
        - *SAST*[*k*,*i*,*r*] = SAST[*k-1*,*i*,*r*] + *d*[*s*,*i*,*r*]/*ts*
        - *ASAST*[*k*,*i*,*r*] = *SAST*[*k*,*i*,*r*] – *ato*
        - *SAET*[*k*,*i*,*r*] = SAST[*k*,*i*,*r*] + *TSB* + *d*[*s*,*i*,*r*] /*ts*
        - *SD*[*k*,*i*,*r*] = *d*[*s*,*i*,*r*] /*ts*

- if **SegmentTemplate**`@media` contains `$Number$`
  - *Address* = *Address* + 1
  - *URL*[*k*,*i*,*r*] = `URLTemplate` (`$Number$`, *Address*)

  else
  - Address = Address + *d*[*s*,*i*,*r*]
  - *URL*[*k*,*i*,*r*] = `URLTemplate` (`$Time$`, *Address*)
- k2[*i*,*r*] = *k*
- *SAET*[*0*,*i*,*r*] = *SAET*[k2[*i*,*r*],*i*,*r*]

Note that not all segments documented above may necessarily be accessible at time *NOW*, but only those that are within the segment availability time window.

Hence, the number of the first media segment described in the MPD for this Period, k1[*i*,*r*], is the smallest *k*=1, 2, ... for which *SAST*[*k*,*i*,*r*] >= *NOW*.

The latest available Period *i*[*NOW*] is the Period *i* with the largest *PEwc*[*i*] and *PEwc*[*i*] is smaller than or equal to *NOW*.

The latest available segment *k*[*NOW*] available for a Representation of Period *i*[*NOW*] (also the live edge segment) is the segment with the largest *k*=0,1,2,... such that *SAST*[*k*,*i*,*r*] is smaller than or equal to *NOW*. Note that this contains the Initialization Segment with *k*=0 as not necessarily any media segment may yet be available for Period *i*[*NOW*]. In this case, last media segment k2[*i*[*NOW*]-1,*r*], i.e., the last media segment of the previous Period is the latest accessible media Segment.

However, if the `@availabilityTimeOffset` is present, then the segments for this Representation are available earlier than the nominal segment availability start time, namely at *ASAST*[*k*,*i*,*r*].

4.3.2.2.8. URL Generation with Segment Template

The function URL Template function `URLTemplate`(`ReplacementString`, Address) generates a URL. For details refer to ISO/IEC 23009-1 [1], section 5.3.9.4. Once the Segment is generated, processing of the Base URLs that apply on this segment level is done as defined in ISO/IEC 23009-1, section 5.6.

4.3.2.2.9. Synchronized Playout and Seamless Switching

In order to achieve synchronized playout across different Representations, typically from different Adaptation Sets, the different Representations are synchronized according to the presentation time in the Period. Specifically, the earliest presentation time of each Segment according to section 4.3.2.2.6 determines the playout of the Segment in the Period and therefore enables synchronized playout of different media components as well as seamless switching within one media component.

4.3.3. Service Offering Requirements and Guidelines

4.3.3.1. General Service Offering Requirements

For dynamic service offerings, the MPD shall conform to DASH-IF IOP as defined in section 3 and shall at least contain the mandatory information as documented in Table 6.

If such an MPD is accessible at time *NOW* at the location **`MPD.Location`**, then

- all Segments for all Representations in all Periods as announced in an MPD shall be available latest at the announced segment availability start time *SAST*[*k*,*i*,*r*] at all *URL*[*k*,*i*,*r*] as derived in section 4.3.2.2;
- all Segments for all Representations in all Periods as announced in an MPD shall at least be available until the announced segment availability end time *SAET*[*k*,*i*,*r*] at all *URL*[*k*,*i*,*r*] as derived in section 4.3.2.2;
- for all Media Segments for all Representations in all Periods as announced in an MPD the Segment in this Period is available prior to the sum of Period start, earliest presentation time and segment duration, i.e. *SAST*[*k*,*i*,*r*] <= *PSwc*[*i*] + *SD*[*k*,*r*,*i*] + *EPT*[*k*,*r*,*i*];
- if a Media Segments with segment number *k* is delivered over a constant bitrate channel with bitrate equal to value of the `@bandwidth` attribute then each presentation time *PT* is available at the client latest at time with a delay of at most *PT* + *MBT*.

4.3.3.2. Dynamic Service Offering Guidelines

4.3.3.2.1. Introduction

In order to offer a simple dynamic service for which the following details are known in advance,

- start at wall-clock time `START`,
- exact duration of media presentation `PDURATION`,
- location of the segments for each Representation at " `http://example.com/$RepresentationID$/$Number$`",

a service provide may offer an MPD as follows:

Table 7 – Basic Service Offering

MPD Information	Value
MPD@type	dynamic
MPD@availabilityStartTime	START
MPD@mediaPresentationDuration	PDURATION
MPD@suggestedPresentationDelay	SPD
MPD@minBufferTime	MBT
MPD@timeShiftBufferDepth	TSB
MPD.BaseURL	"http://example.com/"
Period@start	PSTART
Representation@bandwidth	BW
SegmentTemplate@media	"$RepresentationID$/$Number$"
SegmentTemplate@startNumber	1
SegmentTemplate@duration	SDURATION

Note that the setting of capitalized parameters is discussed in section 4.3.3.2.2.

According to the work-flow shown in Annex B:

- the MPD is generated and published prior to time START such that DASH clients may access it prior to the start of the Media Presentation.
- no redundant tools are considered.
- the encoder and the segmenter generate segments of duration SDURATION and publish those on the origin server, such that they are available at URL[*k*] latest at their announced segment availability start time SAST[*k*].

Based on the details in section 4.3.2.2, the Segment Information is derived as:

- k1 = 1
- k2 = ceil(PDURATION/SDURATION)
- for k = 1, ..., k2
    - SAST[*k*] = START + PSTART + *k**SDURATION
    - SAET[*k*] = SAST[*k*] + TSB + SDURATION
    - SD[*k*] = SDURATION
    - URL[*k*] = http://example.com/$RepresentationID$/k
- The segment availability times of the Initialization Segment are as follows:
    - SAST[*0*] = START + PSTART
    - SAET[0] = SAET[k2]

4.3.3.2.2. Basic Parameter Settings

In the following recommendations are provided for the

- Time Shift Buffer Depth (TSB):
    - If the content should be consumed at the live edge, then the time shift buffer depth should be set short. However, the TSB should not be smaller than the recommended value of 4*SDURATION and 6 seconds in media time in order for the client to do some prebuffering in more difficult network conditions.
    - If no restrictions on the accessibility of the content are provided, then the TSB may be set to a large value that even exceeds PDURATION.
- Suggested Presentation Delay (SPD)
    - If synchronized play-out with other devices adhering to the same rule is desired and/or the service provider wants to define the typical live edge of the program, then this value should be provided. The service provider should set the value taking into account at least the following:
        - the desired end-to-end latency
        - the typical required buffering in the client, for example based on the network condition
        - the segment duration SDURATION
        - the time shift buffer depth TSB
    - A reasonable value may be 2 to 4 times of the segment duration SDURATION, but the time should not be smaller than 4 seconds in order for the client to maintain some buffering.
- Segment Duration (SDURATION)
    - The segment duration typically influences the end-to-end latency, but also the switching and random access granularity as in DASH-264/AVC each segment

starts with a stream access point which can also be used as s switch point. The service provider should set the value taking into account at least the following:
    - the desired end-to-end latency
    - the desired compression efficiency
    - the start-up latency
    - the desired switching granularity
    - the desired amount of HTTP requests per second
    - the variability of the expected network conditions
  - Reasonable values for segment durations are between 1 second and 10 seconds.
- Minimum Buffer Time (`MBT`) and bandwidth (`BW`)
  - the value of the minimum buffer time **does not provide any instructions to the client on how long to buffer the media**. This aspect is covered in 4.3.4.4. The value describes how much buffer a client should have under ***ideal*** network conditions. As such, `MBT` is not describing the burstiness or jitter in the network, it is describing the burstiness or jitter in the **content encoding**. Together with the `BW` value, it is a property of the content. Using the "leaky bucket" model, it is the size of the bucket that makes `BW` true, given the way the content is encoded.
  - The minimum buffer time provides information that for each Stream Access Point (and in the case of DASH-IF therefore each start of the Media Segment), the property of the stream: If the Representation (starting at any segment) is delivered over a constant bitrate channel with bitrate equal to value of the `BW` attribute then each presentation time *PT* is available at the client latest at time with a delay of at most *PT + MBT*.
  - In the absence of any other guidance, **the** `MBT` **should be set** to the maximum GOP size (coded video sequence) of the content, which quite often is identical **to the maximum segment duration**. The *MBT* may be set to a smaller value than maximum segment duration, but should not be set to a higher value.

In a simple and straightforward implementation, a DASH client decides downloading the next segment based on the following status information:

- the currently available buffer in the media pipeline, *buffer*
- the currently estimated download rate, *rate*
- the value of the attribute `@minBufferTime`, *MBT*
- the set of values of the `@bandwidth` attribute for each Representation i, *BW*[*i*]

The task of the client is to select a suitable Representation *i*.

The relevant issue is that starting from a SAP on, the DASH client can continue to playout the data. This means that at the current time it does have *buffer* data in the buffer. Based on this model the client can download a Representation *i* for which $BW[i] \leq rate*buffer/MBT$ without emptying the buffer.

Note that in this model, some idealizations typically do not hold in practice, such as constant bitrate channel, progressive download and playout of Segments, no blocking and congestion of other HTTP requests, etc. Therefore, a DASH client should use these values with care to compensate such practical circumstances; especially variations in download speed, latency, jitter, scheduling of requests of media components, as well as to address other practical circumstances.

One example is if the DASH client operates on Segment granularity. As in this case, not only parts of the Segment (i.e., MBT) needs to be downloaded, but the entire Segment, and if the MBT is smaller than the Segment duration, then rather the segment duration needs to be used instead of the MBT for the required buffer size and the download scheduling, i.e. download a Representation *i* for which $BW[i] \leq rate*buffer/max\_segment\_duration$.

For low latency cases, the above parameters may be different.

4.3.3.2.3. Example

Assume a simple example according to Table 10.

Table 8 – Basic Service Offering

MPD Information	Value
MPD@type	dynamic
MPD@availabilityStartTime	START
MPD@mediaPresentationDuration	43sec
MPD@suggestedPresentationDelay	15sec
MPD@minBufferTime	5sec
MPD@timeShiftBufferDepth	25sec
MPD.BaseURL	"http://example.com/"
Period@start	0
SegmentTemplate@media	"$RepresentationID$/$Num-ber$"
SegmentTemplate@startNumber	1
SegmentTemplate@duration	5sec

Based on the derivation in section 4.3.3.2.1, the following holds:

- k1 = 1, k2 = 9
- for k = 1, ..., k2
  - SAST[*k*] = START + *k**5sec
  - SAET[*k*] = SAST[*k*] + 30sec
  - URL[*k*] = http://example.com/1/k
- The segment availability times of the Initialization Segment are as follows:
  - SAST[*0*] = START
  - SAET[0] = START + 75 sec

Figure 5 shows the availability of segments on the *server* for different times NOW. In particular, before START no segment is available, but the segment URLs are valid. With time *NOW* advancing, segments get available.

Figure 5 Segment Availability on the Server for different time NOW (blue = valid but not yet available segment, green = available Segment, red = unavailable Segment)

4.3.3.3. Content Offering with Periods

4.3.3.3.1. General

For content offered within a Period, and especially when offered in multiple Periods, then the content provider should offer the content such that actual media presentation time is as close as possible to the actual Period duration. It is recommended that the Period duration is the maximum of the presentation duration of all Representations contained in the Period.

A typical Multi-Period Offering is shown in Table 9. This may for example represent a service offering where main content provided in Period 1 and Period 3 are interrupted by an inserted Period 2.

Table 9 Multi-Period Service Offering

MPD Information	Value
MPD@type	dynamic
MPD@availabilityStartTime	START
MPD@mediaPresentationDuration	PDURATION
MPD@suggestedPresentationDelay	SPD
MPD@minBufferTime	MBT
MPD@timeShiftBufferDepth	TSB
MPD.BaseURL	"http://example.com/"
Period@start	PSTART
SegmentTemplate@media	"1/$RepresentationID$/$Number$"

SegmentTemplate@startNumber	1
SegmentTemplate@duration	SDURATION1
Period@start	PSTART2
Representation@availabilityTimeOffset	ATO2
SegmentTemplate@media	"2/$RepresentationID$/$Number$"
SegmentTemplate@startNumber	1
SegmentTemplate@duration	SDURATION2
Period@start	PSTART3
SegmentTemplate@media	"1/$RepresentationID$/$Number$"
SegmentTemplate@startNumber	STARTNUMBER2
SegmentTemplate@duration	SDURATION1
SegmentTemplate@presentationTimeOffset	PTO

The work flow for such a service offering is expected to be similar to the one in section 4.3.2.2.1.

Based on the details in section 4.3.2.2, the Segment Information is derived as:

- Period 1
    - PSwc[1] = `START` + `PSTART`
    - PEwc[1] = `START` + `PSTART2`
    - k1 = 1
    - k2 = `ceil((PSTART2-PSTART1)/SDURATION)`
    - for k = 1, ..., k2
        - SAST[*k*] = PSwc[1] + *k**`SDURATION`
        - SAET[*k*] = SAST[*k*] + `TSB` + `SDURATION`
        - SD[*k*] = `SDURATION`
        - URL[*k*] = `http://example.com/1/$RepresentationID$/k`
    - SAST[0] = PSwc[1]
    - SAET[0] = SAET[k2]
- Period 2
    - PSwc[2] = `START` + `PSTART2`
    - PEwc[2] = `START` + `PSTART3`
    - k1 = 1
    - k2 = `ceil((PSTART3-PSTART2)/SDURATION2)`
    - for k = 1, ..., k2
        - SAST[*k*] = PSwc[2] + *k**`SDURATION2`
        - ASAST[*k*] = SAST[*k*] – `ATO2`
        - SAET[*k*] = SAST[*k*] + `TSB` + `SDURATION2`
        - SD[*k*] = `SDURATION2`
        - URL[*k*] = `http://example.com/2/$RepresentationID$/k`
    - SAST[0] = PSwc[2]
    - SAET[0] = SAET[k2]

- Period 3
  - PSwc[3] = `START` + `PSTART3`
  - PEwc[3] = `START` + `PDURATION`
  - k1 = 1
  - k2 = `ceil((PDURATION-PSTART3)/SDURATION1)`
  - for k = 1, ..., k2
    - SAST[*k*] = PSwc[3] + *k**`SDURATION1`
    - SAET[*k*] = SAST[*k*] + `TSB` + `SDURATION1`
    - SD[*k*] = `SDURATION1`
    - URL[*k*] = "`http://example.com/1/$RepresentationID$/(k+STARTNUMBER2-1)`"
  - SAST[0] = PSwc[3]
  - SAET[0] = SAET[k2]

Note that the number *k* describes position in the Period. The actual number used in the segment template increased by the one less than the actual start number.

In order to ensure that the attribute **`Period`**`@start` can accurately document the duration of the previous Period and to avoid that the player may fall into a loop searching for a Segment in the wrong Period, it is recommended to accurately document the **`Period`**`@start` time. In order to fulfill this, it is recommended to use video track time scale to document the exact duration of the Period. A media time scale of at most 90 kHz is recommended and may be represented by the `xs:duration` type of **`Period`**`@start`.

4.3.3.3.2. Continuous Period Offering

Continuous Period offering as defined in section 3.2.12 may be used. If multiple periods are offered primarily for robustness or MPD changes, the continuous period should be used by the content author to provide seamless experiences for the user. If the condition of continuous timelines is not fulfilled, but all other conditions, then period-connectivity may be used as defined in section 3.2.12.

4.3.3.4. Content Offering with Segment Timeline

4.3.3.4.1. Basic Operation

In order to offer a dynamic service that takes into account

- variable segment durations
- gaps in the segment timeline of one Representation,

the Segment timeline as defined in ISO/IEC 23009-1, section 5.3.9.6 may be used as an alternative to the `@duration` attribute as shown in section 4.3.3.2.

Table 10 – Service Offering with Segment Timeline

MPD Information	Value
`MPD``@type`	`dynamic`
`MPD``@availabilityStartTime`	`START`
`MPD``@mediaPresentationDuration`	`PDURATION`

MPD@suggestedPresentationDelay	SPD
MPD@minBufferTime	MBT
MPD@timeShiftBufferDepth	TSB
MPD.BaseURL	"http://example.com/"
Period@start	PSTART
SegmentTemplate@media	"$RepresentationID$/$Number$"
SegmentTemplate@startNumber	1
SegmentTemplate.SegmentTimeline	t[i], n[i], d[i], r[i]

According to the work-flow shown in Annex B:

- the MPD is generated and published prior to time START such that DASH clients may access it prior to the start of the Media Presentation.
- no redundant tools are considered.
- the encoder and the segmenter generally should generate segments of constant duration SDURATION and publish those on the origin server, such that they are available at URL[*k*] latest at their announced segment availability start time SAST[*k*]. However, the server may offer occasional shorter segments for encoding optimizations, e.g. at scene changes, or segment gaps (for details see section 6). If such an irregular segment is published the MPD needs to document this by a new **S** element in the segment timeline.

If the segment timeline is used and the $Time$ template is used, then the times in the MPD shall accurately present media internal presentation times.

If the segment timeline is and the $Number$ template is used, then the MPD times shall at most deviate from the earliest presentation time documented in the MPD by 0.5sec.

Based on these considerations, it is not feasible to operate with a single MPD if the content is not yet known in advance. However, pre-prepared content based on the segment timeline may be offered in a dynamic fashion. The use of the Segment Timeline is most suitable for the case where the MPD can be updated. For details refer to section 4.4.

4.3.3.4.2. Basic Parameter Settings

The parameters for TSB and SPD should be set according to section 4.3.3.2.2. The segment duration SDURATION may be set according to section 4.3.3.2.2, but it should be considered that the service provider can offer shorter segments occasionally.

4.3.3.5. Joining Recommendation

By default, an MPD with **MPD**@type="dynamic" suggests that the client would want to join the stream at the live edge, therefore to download the latest available segment (or close to, depending on the buffering model), and then start playing from that segment onwards.

However there are circumstances where a dynamic MPD might be used with content intended for playback from the start, or from another position. For example, when a content provider offers ‘start again’ functionality for a live program, the intention is to make the content available as an on-demand program, but not all the segments will be available immediately.

This may be signalled to the DASH client by including an MPD Anchor, with either

- the `t` parameter, or
- both the `period` and `t` parameter, in the MPD URL provided to the DASH client, or
- the `POSIX` parameter, for details refer to Amd.3 of ISO/IEC 23009-1:2014 [4][4].

The format and behaviour of MPD Anchors is defined in section C.4 of ISO/IEC 23009-1. Specifically the `POSIX` parameter is defined in Amd.3 of ISO/IEC 23009-1:2014 [4].

For example to start from the beginning of the MPD the following would be added to the end of the MPD URL provided to the DASH client:

```
#t=0
```

Or to start from somewhere other than the start, in this case 50 minutes from the beginning of the period with Period ID "`program_part_2`":

```
#period=program_part_2&t=50:00
```

Starting from a given UTC time can be achieved using the POSIX clock with t parameter. For example, starting playback from Wed, 08 Jun 2016 17:29:06 GMT would be expressed as

`#t=posix:1465406946`

`#t=posix:now` stands for "live edge"

Notes:

- as per section C.4 of ISO/IEC 23009-1 the time indicated using the t parameter is as per the field definition of the W3C Media Fragments Recommendation v1.0 section 4.2.1.
- the period ID has to be URL encoded/decoded as necessary and needs to match one of the **`Period`**`@id` fields in the MPD.

Where an MPD Anchor is used it should refer to a time for which segments are currently available in the MPD.

4.3.4. Client Operation, Requirements and Guidelines

4.3.4.1. Basic Operation for Single Period

A DASH client is guided by the information provided in the MPD. A simple client model is shown in Figure 6.

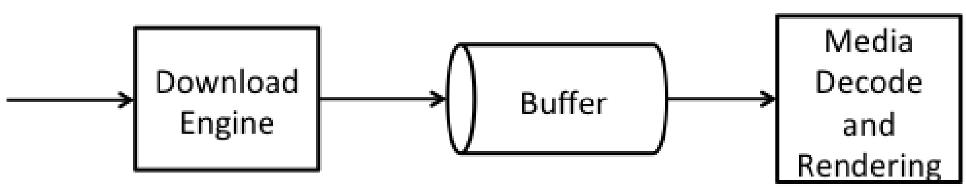

Figure 6 Simple Client Model

Assume that the client has access to an MPD and the MPD contains the parameters in Table 6, i.e. it consumes a dynamic service with fixed media presentation duration.

In addition in the following for simplicity it is assumed that the MPD only contains a single Period with period start time *PSwc*[*i*] and the MPD-URL does not include any fragment parameters according to section 4.3.3.5.

The following example client behavior may provide a continuous streaming experience to the user:

1) The client parses the MPD, selects a collection of Adaptation Sets suitable for its environment based on information provided in each of the **`AdaptationSet`** elements.
2) Within each Adaptation Set it selects one Representation, typically based on the value of the `@bandwidth` attribute, but also taking into account client decoding and rendering capabilities.
3) The client creates a list of accessible Segments at least for each selected Representation taking into account the information in the MPD as documented in Table 6 and the current time *JOIN* in the client and in particular the segment closest to the live edge referred to the *live edge segment*. For details refer to section 4.3.4.2.
4) The client downloads the initialization segment of the selected Representations and then accesses the content by requesting entire Segments or byte ranges of Segments. Typically at any time the client downloads the next segment at the larger of the two: (i) completion of download of current segment or (ii) the Segment Availability Start Time of the next segment. Note that if the `@availabilityTimeOffset` is present, then the segments may be downloaded earlier, namely at the adjusted segment availability start time. Based on the buffer fullness and other criteria, rate adaptation is considered. Typically the first media segment that is downloaded is the *live edge segment*, but other decisions may be taken in order to minimize start-up latency. For details on initial buffering, refer to section 4.3.4.4.
5) According to Figure 6 media is fed into buffer and at some point in time, the decoding and rendering of the media is kicked off. The downloading and presentation is done for the selected Representation of each selected Adaptation. The synchronization is done using the presentation time in the Period as documented in section 4.3.2.2.9. For synchronized playout, the exact presentation times in the media shall be used.

   Once presentation has started, the playout process is continuous. The playout process expects media to be present in the buffer continuously. If the **`MPD`**`@suggestedPresentationDelay` is present, then this value may be used as the presentation delay *PD*. If the **`MPD`**`@suggestedPresentationDelay` is not present, but the client is expected to consume the service at the live edge, then a suitable presentation delay should be selected, typically between the value of `@minBufferTime` and the value of `@timeShiftBufferDepth`. It is recommended that the client starts rendering the first sample of the downloaded media segment *k* with earliest presentation time EPT(*k*) at *PSwc*[*i*] + (EPT(*k*) - *o*[*r*,*i*]) + *PD*. For details on selecting and minimizing end-to-end latency as well as the start-up latency, see section 4.3.4.4.
6) The client may request Media Segments of the selected Representations by using the generated Segment list during the availability time window.

7) Once the presentation has started, the client continues consuming the media content by continuously requesting Media Segments or parts of Media Segments and playing content that according to the media presentation timeline. The client may switch Representations taking into updated information from its environment, e.g. change of observed throughput. In a straight-forward implementation, with any request for a Media Segment starting with a stream access point, the client may switch to a different Representation. If switching at a stream access point, the client shall switch seamlessly at such a stream access point.
8) With the wall-clock time *NOW* advancing, the client consumes the available Segments. As *NOW* advances the client possibly expands the list of *available* Segments for each Representation in the Period according to the procedures specified in 4.3.4.2.
9) Once the client is consuming media contained in the Segments towards the end of the announced media in the Representation, then either the Media Presentation is terminated, a new Period is started (see subsection 4.3.4.3) or the MPD needs to be refetched. For details on MPD updates and refetching, please refer to section 4.4.

4.3.4.2. Determining the Segment List

For a single Period content the client determines the available Segment List at time *NOW* according to section 4.3.2.2.7 taking into account the simplified offering in Table 7 as

- k1 = 1
- k2 = `ceil(PDURATION/SDURATION)`
- SAST[*k*] = `START` + `PSTART` + *k**`SDURATION` for k = 0, 1, ..., k2
- ASAST[*k*] = SAST[*k*] - `ATO`
- SAET[*k*] = SAST[*k*] + `TSB` + `SDURATION` for k = 1, ..., k2
- SAET[0] = SAET[k2]
- SD[*k*] = `SDURATION`
- URL[*k*] = `http://example.com/$RepresentationID$/k`
- k[*NOW*] = MIN(`floor` ((*NOW* - `START` - `PSTART`)/`SDURATION`), k2)
- k*[*NOW*] = MAX(k1, `floor`((*NOW* - `START` - `PSTART` - `TSB`)/`SDURATION`)

Note that if k[*NOW*] is *0*, then only the Initialization Segment is available. The live edge segment if provided as k[*NOW*]. If the `@availabilityTimeOffset` is present, then the segments for this Representation may be downloaded from ASAST[*k*] onwards.

4.3.4.3. Multi-Period Content

In an extension to the description in section 4.3.4.1 assume now that the client has access to an MPD and the MPD contains content with multiple Periods, for example following the parameters in Table 9. The start time of each Period is computed as period start time *PSwc*[*i*]. and the MPD-URL does not include any fragment parameters according to section 4.3.3.5.

In an extension of bullet 3 in section 4.3.4.1,

the client creates a list of accessible Segments at least for each selected Representation taking into account the information in the MPD as documented in Table 6 and the current time *NOW* in the client and in particular the segment closest to the live edge referred to the *live edge segment*.

For this it needs to take into account the latest Period *i*[*NOW*]. The latest Period and the latest segment are obtained as follows with i* the index of the last Period.:

- if *NOW* <= PSwc[1]
  - no segment is yet available
- else if *NOW* > PSwc[i*]
  - the last one and the latest segment is available is k2[i*]
- else if *NOW* > PSwc[i*] + `TSB`
  - no segment is available any more
- else if PSwc[1] < *NOW* <= PEwc[i*]
    - *i'* the such that PSwc[*i'*] < *NOW* <= PEwc[*i'*]
    - k[*NOW*] = MIN(`floor` ((*NOW* - PEwc[*i'*] - PSwc[*i'*])/`SDURATION[i']`), k2)
- Note again that if k[*NOW*] is *0*, then only the Initialization Segment is available. If the Period is not the first one, then the last available Media Segment is the last Media Segment of the previous Period.

In an extension of bullet 9 in section 4.3.4.1,

the client consumes media in one Period. Once the client is consuming media contained in the Segments towards the end of the announced media in the Representation, and the Representation is contained not in the last Period, then the DASH clients generally needs to reselect the Adaptation Sets and a Representation in same manner as described in bullet 1 and 2 in section 4.3.4.1. Also steps 3, 4, 5 and 6 need to be carried out at the transition of a Period. Generally, audio/video switching across period boundaries may not be seamless. According to ISO/IEC 23009-1, section 7.2.1, at the start of a new Period, the playout procedure of the media content components may need to be adjusted at the end of the preceding Period to match the *PeriodStart* time of the new Period as there may be small overlaps or gaps with the Representation at the end of the preceding Period. Overlaps (respectively gaps) may result from Media Segments with actual presentation duration of the media stream longer (respectively shorter) than indicated by the Period duration. Also in the beginning of a Period, if the earliest presentation time of any access unit of a Representation is not equal to the presentation time offset signalled in the `@presentationTimeOffset`, then the playout procedures need to be adjusted accordingly.

The client should play the content continuously across Periods, but there may be implications in terms of implementation to provide fully continuous and seamless playout. It may be the case that at Period boundaries, the presentation engine needs to be reinitialized, for example due to changes in formats, codecs or other properties. This may result in a re-initialization delay. Such a re-initialization delay should be minimized. If the Media Presentation is of type dynamic, the addition of the re-initialisation delay to the playout may result in drift between the encoder and the presentation engine. Therefore, the playout should be adjusted at the end of each Period to provide a continuous presentation without adding drift between the time documented in the MPD and the actual playout, i.e. the difference between the actual playout time and the Period start time should remain constant.

If the client presents media components of a certain Adaptation Set in one Period, and if the following Period has assigned an identical Asset Identifier, then the client should identify an associated Period and, in the absence of other information, continue playing the content in the associated Adaptation Set.

If furthermore the Adaptation Sets are period-continuous, i.e. the presentation times are continuous and this is signalled in the MPD, then the client shall seamlessly play the content across the Period boundary under the constraints in section 4.3.3.3.2. The presentation time offset should be ignored. Most suitably the client may continue playing the Representation in the Adaptation Set with the same `@id`, but there is no guarantee that this Representation is available. In this case the client shall seamlessly switch to any other Representation in the Adaptation Set.

If otherwise the Adaptation Sets are period-connected and this is signaled in the MPD, then the client should avoid re-initializing media decoders. The client should inform the media decoder on a timeline discontinuity obeying the value of `@presentationTimeOffset` attribute, but it may continue processing the incoming Segments without re-initializing the media decoder. The presentation time offset should be used to seamlessly play the content across the Period boundary under the constraints in section 4.3.3.3.2.

4.3.4.4. Joining, Initial Buffering and Playout Recommendations

4.3.4.4.1. General

A DASH client should start playout from:

- The time indicated by the MPD Anchor, if one is present
- The live edge, if there is no MPD Anchor and **MPD**`@type="dynamic"`.

4.3.4.4.2. Joining at the live edge

For joining at the live edge there are basically two high-level strategies:

- Every client participating in the service commits to the same presentation delay (PD) relative to the announced segment availability start time at start-up and in continuous presentation, possible using one suggested by the Content Provider and then attempts to minimise start-up latency and maintain the buffer. The content provider may have provided the **MPD**`@suggestedPresentationDelay` (`SPD`) or may have provided this value by other means outside the DASH formats. The content author should be aware that the client may ignore the presence of **MPD**`@suggestedPresentationDelay` and may choose its own suitable playout scheduling.

- The client individually picks the presentation delay (PD) in order to maximize stable quality and does this dependent on its access, user preferences and other considerations.

In both cases the client needs to decide, which segment to download first and when to schedule the playout of the segment based on the committed PD.

A DASH client would download an available segment and typically render the earliest presentation time EPT(k) of the segment at $PSwc[i] + (\text{EPT}(k) - o[r,i]) + PD$. As PD may be quite large, for example in order to provision for downloading in varying bitrate conditions, and if a segment is downloaded that was just made available it may result in larger start up delay.

Therefore, a couple of strategies may be considered as a tradeoff of for start-up delay, presentation delay and sufficient buffer at the beginning of the service, when joining at the live edge:

1. The client downloads the next available segment and schedules playout with delay `PD`. This maximizes the initial buffer prior to playout, but typically results in undesired long start-up delay.
2. The client downloads the latest available segment and schedules playout with delay `PD`. This provides large initial buffer prior to playout, but typically results in undesired long start-up delay.
3. The client downloads the earliest available segment that can be downloaded to schedules playout with delay `PD`. This provides a smaller initial prior to playout, but results in reasonable start-up delay. The buffer may be filled gradually by downloading later segments faster than their media playout rate, i.e. by initially choosing Representations that have lower bitrate than the access bandwidth.

In advanced strategies the client may apply also one or more of the following:

1. Actual rendering may start not with the sample of the earliest presentation time, but the one that matches as closely as possible $PSwc[i]$ + (PT - $o[r,i]$) + PD equal to *NOW*.
2. The client may start rendering even if only a segment is downloaded partially.

Also if the `@availabilityTimeOffset` is present and the segment has an adjusted segment availability start time, then the segments may be downloaded earlier.

4.3.4.5. Requirements and Recommendations

In summary, a client that access a dynamic MPD shall at least obey the following rules:

- The client shall be able to consume single Period and multi-Period content
- If multi-period content is offered in a seamless manner, the client shall play seamlessly across Period boundaries.

4.3.5. Additional DVB-DASH alignment aspects

For alignment with DVB-DASH [42], the following should be considered:

- Reasonable requirements on players around responding to response codes are provided in DVB DASH in section 10.8.6.
- Further guidelines on live edge aspects are provided in DVB DASH section 10.9.2.

DVB DASH also provides recommendations in order to apply weights and priorities to different networks in a multi Base URL offering in section 10.8.2.1.

4.3.6. Considerations on live edge

Detecting the live edge segment in DASH as well as providing a sanity check for the MPD author on the correctness of the offering may be achieved for example by the following means:

- If the MPD contains a `@publishTime` attribute with value `PUBT`, then at the publication of the MPD all Segments according to the computation in section 4.3.4.2 and 4.3.4.3 with *NOW* set to `PUBT` shall be available.

- If the MPD contains a `@publishTime` attribute with value `PUBT` and a Representation contains a Segment timeline with the `@r` attributed of the last **S** element being non-negative, then the last Segment describe in this Segment timeline shall have a Segment availability start time smaller than `PUBT` and the sum of the segment duration and the segment availability start time shall be larger than `PUBT`.

A DASH client should avoid being too aggressive in requesting segments exactly at the computed segment availability start time, especially if it is uncertain to be fully synchronized with the server. If the DASH client observes issues, such as 404 responses, it should back up slightly in the requests.

In addition, for a content authoring to avoid too aggressive requests and possible 404 responses, the content author may schedule the segment availability start time in the MPD with a small safety delay compared to the actual publish time. This also provides the content author a certain amount of flexibility in the publishing of Segments. However, note that such safety margins may lead to slightly increased end-to-end latencies, so it is a balance to be taken into account.

4.4. Simple Live Service Offering including MPD Updates

4.4.1. Background and Assumptions

If many cases, the service provider cannot predict that an MPD that is once offered, may be used for the entire Media Presentations. Examples for such MPD changes are:

- The duration of the Media Presentation is unknown
- The Media Presentation may be interrupted for advertisements which requires proper splicing of data, for example by adding a Period
- Operational issues require changes, for example the addition of removal of Representations or Adaptation Sets.
- Operational problems in the backend, for example as discussed in section 4.8.
- Changes of segment durations, etc.

In this case the MPD typically only can describe a limited time into the future. Once the MPD expires, the service provider expects the client to recheck and get an updated MPD in order to continue the Media Presentation.

The main tool in MPEG-DASH is Media Presentation Description update feature as described in section 5.4 of ISO/IEC 23009-1. The MPD is updated at the server and the client is expected to obtain the new MPD information once the determined Segment List gets to an end.

If the MPD contains the attribute **MPD**`@minimumUpdatePeriod`, then the MPD in hand will be updated.

According to the clustering in section 4.2, we distinguish two different types of live service offerings:

- MPD controlled live service offering: In this case the DASH client typically frequently polls the MPD update server whether an MPD update is available or the existing MPD can still be used. The update frequency is controlled by MPD based on the attribute **MPD**`@minimumUpdatePeriod`. Such a service offering along with the client procedures is shown in section 4.4.2.

- MPD and segment controlled offerings. In this case the DASH client needs to parse segments in order to identify MPD validity expirations and updates on the MPD update server. MPD expiry events as described in section 5.10 of ISO/IEC 23009-1 "are pushed" to the DASH client as parts of downloaded media segments. This offering along with the client procedures is shown in section 4.5.

This section describes the first type of offering. In section 4.5 the MPD and segment controlled offerings are described. Under certain circumstances a service offering may be provided to both types of clients. An overview how such a service offering may be generated is shown in Annex A.

4.4.2. Preliminaries

4.4.2.1. MPD Information

As the MPD is typically updated over time on the server, the MPD that is accessed when joining the service as well as the changes of the MPD are referred to as MPD instances in the following. This expresses that for the same service, different MPDs exist depending on the time when the service is consumed.

Assume that an MPD instance is present on the DASH server at a specific wall-clock time *NOW*. For an MPD-based Live Service Offering, the MPD instance may among others contain information as available in Table 11. Information included there may be used to compute a list of announced Segments, Segment Availability Times and URLs.

Table 11 – Information related to Live Service Offering with MPD-controlled MPD Updates

MPD Information	Status	Comment
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 except for the case where the **MPD**@mediaPresentationDuration is present. However, such an MPD falls then in an instance as documented in section 4.3.
Period@start	mandatory	the start time of the Period relative to the MPD availability start time. The value is denoted as *PS*.
SegmentTemplate@media	mandatory	the template for the Media Segment

SegmentTemplate@startNumber	optional default	the number of the first segment in the Period. The value is denoted as *SSN*.
SegmentTemplate@duration	exactly one of **SegmentTemplate**@duration or **SegmentTemplate.SegmentTimeline** must be present	the duration of each Segment in units of a time. The value divided by the value of @timescale is denoted as *MD*[*k*] with k=1, 2, ... The segment timeline may contain some gaps.
SegmentTemplate.SegmentTimeline		

4.4.2.2. Segment Information Derivation

Based on an MPD instance including information as documented in Table 11 and available at time *NOW* on the server, a DASH client may derive the information of the list of Segments for each Representation in each Period.

If the Period is the last one in the MPD and the **MPD**@minimumUpdatePeriod is present, then the time *PEwc*[*i*] is obtained as the sum of *NOW* and the value of **MPD**@minimumUpdatePeriod.

Note that with the MPD present on the server and *NOW* progressing, the Period end time is extended. This issue is the only change compared to the segment information generation in section 4.3.2.2.

4.4.2.3. Some Special Cases

If the **MPD**@minimumUpdatePeriod is set to 0, then the MPD documents all available segments on the server. In this case the @r count may be set accurately as the server knows all available information.

4.4.3. Service Offering Requirements and Guidelines

4.4.3.1. General

The same service requirements as in section 4.3.3.1 hold for any time *NOW* the MPD is present on the server with the interpretation that the Period end time *PEwc*[*i*] of the last Period is obtained as the sum of *NOW* and the value of **MPD**@minimumUpdatePeriod.

In order to offer a simple live service with unknown presentation end time, but only a single Period and the following details are known in advance,

- start at wall-clock time START,
- location of the segments for each Representation at " http://example.com/$RepresentationID$/$Number$",

a service provider may offer an MPD with values according to Table 12.

Table 12 – Basic Service Offering with MPD Updates

MPD Information	Value
MPD@type	dynamic

MPD@availabilityStartTime	START
MPD@publishTime	PUBTIME1
MPD@minimumUpdatePeriod	MUP
MPD.BaseURL	"http://example.com/"
Period@start	PSTART
SegmentTemplate@media	"$RepresentationID$/$Number$"
SegmentTemplate@startNumber	1
SegmentTemplate@duration	SDURATION

According to the work-flow shown in Annex B,

- the MPD is generated and published prior to time START such that DASH clients may access it prior to the start of the Media Presentation. The MPD gets assigned a publish time PUBTIME1, typically a value that is prior to START + PSTART
- no redundant tools are considered.
- the encoder and the segmenter generate segments of duration SDURATION and publish those on the origin server, such that they are available at URL[*k*] latest at their announced segment availability start time SAST[*k*].

Based on the details in section 4.3.2.2 and 4.4.2.2, the Segment Information can be derived at each time *NOW* by determining the end time of the Period *PEwc*[1] = *NOW* + MUP.

The service provider may leave the MPD unchanged on the server. If this is the case the Media Presentation may be terminated with an updated MPD that

- adds the attribute **MPD**@mediaPresentationDuration with value PDURATION
- removes the attribute **MPD**@minimumUpdatePeriod
- changes the **MPD**@publishTime attribute to PUBTIME2

The MPD must be published latest at the end of the Media Presentation minus the value of MUP, i.e. PUBTIME2 <= START + PSTART + PDURATION - MUP.

The minimum update period may also be changed during an ongoing Media Presentation. Note that as with any other change to the MPD, this will only be effective with a delay in media time of the value of the previous MUP.

The principles in this document also holds for multi-period content, for which an MPD update may add a new Period. In the same way as for signalling the end of the Media Presentation, the publish time of the updated MPD with the new period needs to be done latest at the start of the new Period minus the value of the **MPD**@minimumUpdatePeriod attribute of the previous MPD.

Track fragment decode times should not roll over and should not exceed 2^{53} (due to observed limitations in ECMAScript). Two options may be considered:

- the timescale value should be selected that the above mentioned issues are avoided. 32 bit timescales are preferable for installed-base of browsers.

- if large track timescale values are required and/or long-lasting live sessions are setup, this likely requires the use of 64 bit values. Content authors should use 64 bit values for track fragment decode times in these cases, but should not exceed to 2^{53} to avoid truncation issues.

4.4.3.2. Setting the Minimum Update Period Value

Setting the value of the minimum update period primarily affects two main service provider aspects: A short minimum update period results in the ability to change and announce new content in the MPD on shorter notice. However, by offering the MPD with a small minimum update period, the client requests an update of the MPD more frequently, potentially resulting in increased uplink and downlink traffic.

A special value for the minimum update period is 0. In this case, the end time of the period is the current time *NOW*. This implies that all segments that are announced in the MPD are actually available at any point in time. This also allows changing the service provider to offer changes in the MPD that are instantaneous on the media timeline, as the client, prior for asking for a new segment, has to revalidate the MPD.

4.4.3.3. Permitted Updates in an MPD

According to section 5.4 of ISO/IEC 23009-1, when the MPD is updated

- the value of **`MPD`**`@id`, if present, shall be the same in the original and the updated MPD;
- the values of any **`Period`**`@id` attributes shall be the same in the original and the updated MPD, unless the containing **`Period`** element has been removed;
- the values of any **`AdaptationSet`**`@id` attributes shall be the same in the original and the updated MPD unless the containing **`Period`** element has been removed;
- any Representation with the same `@id` and within the same Period as a Representation appearing in the previous MPD shall provide functionally equivalent attributes and elements, and shall provide functionally identical Segments with the same indices in the corresponding Representation in the new MPD.

In addition, updates in the MPD only extend the timeline. This means that information provided in a previous version of the MPD shall not be invalidated in an updated MPD. For failover cases, refer to section 4.8.

In order to make the MPD joining friendly and to remove data that is available in the past, any segments that have fallen out of the time shift buffer may no longer be announced in the MPD. In this case, the Period start may be moved by changing one or both, **`MPD`**`@availabilityStartTime` and **`Period`**`@start`. However, this requires that the `@startNumber`, `@presentationTimeOffset` and **`S`** values need to be updated such that the Segment Information according to section 4.3.2.2.6 is not modified over an MPD update.

If Representations and Adaptations Sets are added or removed or the location of the Segments is changed, it is recommended to update the MPD and provide Adaptation Sets in a period-continuous manner as defined in section 4.3.3.3.2.

4.4.3.4. Usage of Segment Timeline

If the Segment Timeline is used and `@minimumUpdatePeriod` greater than 0, then

- the operation as described in section 4.3.3.4 applies, and for all Representations that use the Segment Timeline:
    - the @r value of the last **S** element of the last regular Period shall be a negative value,
    - only $Number$ template shall be used,
- an MPD may be published for which the additional **S** elements are added at the end. An addition of such **S** element shall be such that clients that have not updated the MPD can still generate the Segment Information based on the previous MPD up to the Period end time. Note that this may lead that such clients have a different segment availability time, but the availability time may be corrected once the MPD is updated.

An example for such an offering is shown in Table 13 where the RVALUE needs to be increased by 1 for each newly published segment.

Table 13 – Service Offering with Segment Timeline and MUP greater than 0

MPD Information	Value
MPD@type	dynamic
MPD@availabilityStartTime	START
MPD@publishTime	PUBTIME1
MPD@minimumUpdatePeriod	MUP > 0
MPD.BaseURL	"http://example.com/"
Period@start	PSTART
SegmentTemplate@media	"$RepresentationID$/$Time$"
SegmentTemplate@d	SDURATION
SegmentTemplate.SegmentTimeline.S@r	-1

4.4.3.5. Last Segment Message

The content author may signal the last segment of a Representation by using the lmsg brand in the segment. If lmsg is signaled in the Representation, the @segmentProfiles attribute for this Representation should signal the 'lmsg' brand for this Representation. If the @segmentProfiles includes the 'lmsg' brand for a Representation, then the 'lmsg' brand shall be included for the last segment of the Representation in a Period.

For non-live MPDs, i.e. @minimumUpdatePeriod not present, and if the lmsg is signaled in the MPD, the DASH client should search for the lmsg brand at at least the last two Segments of a Period, and not request Segments that are later than the one for which the lmsg brand was provided. The player may also parse every Segment for lmsg.

For live MPDs, i.e. @minimumUpdatePeriod is present, if the @segmentProfiles contains the 'lmsg' brand for a certain Representation, then the 'lmsg' brand for signaling the last segment shall be applied for any content with **MPD**@minimumUpdatePeriod present and the **MPD**@type="dynamic".

DASH clients operating based on such an MPD and consuming the service at the live edge typically need to request a new MPD prior to downloading a new segment. However, in order to minimise MPD requests and resulting traffic load, the client may use one or more of the following optimisations:

- If the client fetches the MPD using HTTP, the client should use conditional GET methods as specified in RFC 7232 [23] to reduce unnecessary network usage in the downlink.
- If the `@segmentProfiles` contains the '`lmsg`' brand clients may also rely on the '`lmsg`' message and request a new MPD only in case a segment is received with an '`lmsg`' brand. Otherwise the client may use template constructions to continue determining the URL and the segment availability start time of segments.

If the attribute **MPD**`@minimumUpdatePeriod` is set to a value greater than 0 then all Segments with availability start time less than the sum of the request time and the value of the **MPD**`@minimumUpdatePeriod` will eventually get available at the advertised position at their computed segment availability start time. Note that by providing a **MPD**`@minimumUpdatePeriod` is set to a value greater than 0, DASH servers reduce the polling frequency of clients, but at the same time cannot expect that clients will request an updated MPD to be informed on changes in the segment URL constructions, e.g. at the start of a new Period.

4.4.3.6. Signalling the last segment number in Period

As indicated in clause 4.3.2.2, the content provider may not offer the last segment that is signaled in the MPD. If this is the case, the content provider should signal that the last segment is not the one indicated in the MPD.

At least the following three options may be considered:

- Use the `lmsg` signalling as defined in clause 4.4.3.5.
- Use the Segment Timeline with `@r` value greater or equal to 0.

Add a Supplemental Descriptor with `@schemeIdUri` set to http://dashif.org/guidelines/last-segment-number with the `@value` set to the last segment number.

4.4.4. MPD-based Live Client Operation based on MPD

In an extension to the description in section 4.3.4.1 and section 4.3.4.3, the client now has access to an MPD and the MPD contains the **MPD**`@minimumUpdatePeriod`, for example following the parameters in Table 12. The start time of each Period is computed as period start time *PSwc*[*i*] and the MPD-URL does not include any fragment parameters according to section 4.3.3.5.

The client fetches an MPD with parameters in Table 11 access to the MPD at time `FetchTime`, at its initial location if no **`MPD.Location`** element is present, or at a location specified in any present **`MPD.Location`** element. `FetchTime` is defined as the time at which the server processes the request for the MPD from the client. The client typically should not use the time at which it actually successfully received the MPD, but should take into account delay due to MPD delivery and processing. The fetch is considered successful either if the client obtains an updated MPD or the client verifies that the MPD has not been updated since the previous fetching.

If the client fetches the MPD using HTTP, the client should use conditional GET methods as specified in RFC 7232 [23] to reduce unnecessary network usage in the downlink.

In an extension of bullet 3 in section 4.3.4.1 and section 4.3.4.3

the client creates a list of accessible Segments at least for each selected Representation taking into account the information in the MPD as documented in Table 11 and the current time *NOW* by using the Period end time of the last Period as `FetchTime + MUP`.

In an extension of bullet 9 in section 4.3.4.1 and section 4.3.4.3,

the client consumes media in last announced Period. Once the client is consuming media contained in the Segments towards the end of the announced Period, i.e. requesting segments with segment availability start time close to the validity time of the MPD defined as `FetchTime + MUP`, them, then the DASH client needs to fetch an MPD at its initial location if no **`MPD.Location`** element is present, or at a location specified in any present **`MPD.Location`** element.

If the client fetches the updated MPD using HTTP, the client should use conditional GET methods as specified in in RFC 7232 [23] to reduce unnecessary network usage in the downlink.

The client parses the MPD and generates a new segment list based on the new `FetchTime` and `MUP` of the updated MPD. The client searches for the currently consumed Adaptation Sets and Representations and continues the process of downloading segments based on the updated Segment List.

4.5. MPD and Segment-based Live Service Offering

4.5.1. Preliminaries

4.5.1.1. MPD Information

In order to offer a service that relies on both, information in the MPD and in Segments, the Service Provider may announce that Segments contains inband information. An MPD as shown in Table 9 provides the relevant information. In contrast to the offering in Table 6, the following information is different:

- The **`MPD`**`@minimumUpdatePeriod` is present but is recommended to be set to 0 in order to announce instantaneous segment updates.
- The **`MPD`**`@publishTime` is present in order to identify different versions of MPD instances.
- all Representations of all audio Adaptation Sets or if audio is not present, of all video Adaptation Sets, shall contain an **`InbandEventStream`** element with `@scheme_id_uri = "urn:mpeg:dash:event:2012"` and `@value` either set to 1 or set to 3. The **`InbandEventStream`** element with `@scheme_id_uri = "urn:mpeg:dash:event:2012"` and `@value` either set to 1 or set to 3 may be present in all Representations of all Adaptation Sets.
- **`InbandEventStream`** element with `@scheme_id_uri = "urn:mpeg:dash:event:2012"` and `@value` either set to 1 or set to 3 shall only be signaled on Adaptation Set level.

The information included there may be used to compute a list of announced Segments, Segment Availability Times and URLs.

Table 14 – Service Offering with MPD and Segment-based Live Services

MPD Information	Status	Comment
MPD@type	mandatory, set to "dynamic"	the type of the Media Presentation is dynamic, i.e. Segments get available over time.
MPD@publishTime	mandatory	specifies the wall-clock time when the MPD was generated and published at the origin server. MPDs with a later value of @publishTime shall be an update as defined in 5.4 to MPDs with earlier @publishTime.
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	recommended/mandate to be set to 0 to indicate that frequent DASH events may occur
Period@start	mandatory	the start time of the Period relative to the MPD availability start time. The value is denoted as *PS*.
AdaptationSet.InbandEventStream	mandatory	if the @schemeIdUri is urn:mpeg:dash:event:2014 and the @value is 1, 2 or 3, then this described an Event Stream that supports extending the validity of the MPD.
SegmentTemplate@media	mandatory	the template for the Media Segment
SegmentTemplate@startNumber	optional default	The number of the first segment in the Period. The value is denoted as *SSN*.
SegmentTemplate@duration	exactly one of **SegmentTemplate**@duration or **SegmentTemplate.SegmentTimeline** must be present	the duration of each Segment in units of a time. The value divided by the value of @timescale is denoted as *MD*[*k*] with k=1, 2, ... The
SegmentTemplate.SegmentTimeline		

		segment timeline may contain some gaps.

4.5.1.2. Segment Information Derivation

Based on an MPD instance including information as documented in Table 11 and available at time *NOW* on the server, a DASH client may derive the information of the list of Segments for each Representation in each Period.

If the Period is the last one in the MPD and the **MPD**`@minimumUpdatePeriod` is present, then the time *PEwc*[*i*] is obtained as the sum of *NOW* and the value of **MPD**`@minimumUpdatePeriod`.

Note that with the MPD present on the server and *NOW* progressing, the Period end time is extended. This issue is the only change compared to the segment information generation in section 4.3.2.2.

If the **MPD**`@minimumUpdatePeriod` is set to 0, then the MPD documents all available segments on the server. In this case the `@r` count may be set accurately as the server knows all available information.

4.5.2. Service Offering Requirements and Guidelines

4.5.2.1. Background

In section 5.10 of ISO/IEC 23009-1, section 5.10, DASH events are defined. For service offerings based on the MPD and segment controlled services, the DASH events specified in section 5.10.4 may be used. Background is provided in the following.

DASH specific events that are of relevance for the DASH client are signalled in the MPD. The URN "`urn:mpeg:dash:event:2012`" is defined to identify the event scheme defined in Table 10.

Table 15 `InbandEventStream@``value` attribute for scheme with a value "`urn:mpeg:dash:event:2012`"

`@value`	**Description**
`1`	indicates that MPD validity expiration events as defined in 5.10.4.2 are signalled in the Representation. MPD validity expiration is signalled in the event stream as defined in 5.10.4.2 at least in the last segment with earliest presentation time smaller than the event time.
`2`	indicates that MPD validity expiration events as defined in 5.10.4.3 are signalled in the Representation. MPD validity expiration is signalled in the event stream as defined in 5.10.4.2 at least in the last segment with earliest presentation time smaller than the event time. In addition the message includes an MPD Patch as defined in 5.10.4.3 in the **`message_data`** field.
`3`	indicates that MPD validity expiration events as defined in 5.10.4.3 are signalled in the Representation. MPD validity expiration is signalled in the event stream as defined in 5.10.4.2 at least in the last segment with earliest presentation time smaller than the event time. In addition the message includes a full MPD as defined in 5.10.4.4 in the message_data field.

Note: DVB DASH specification [42] does not include the value 3.

MPD validity expiration events provide the ability to signal to the client that the MPD with a specific publish time can only be used up to a certain media presentation time.

Figure 4 shows an example for MPD validity expiration method. An MPD signals the presence of the scheme in one or several Representations. Once a new MPD gets available, that adds new information not present in the MPD with `@publishTime="2012-11-01T09:06:31.6"`, the expiration time of the current MPD is added to the segment by using the emsg box. The information may be present in multiple segments.

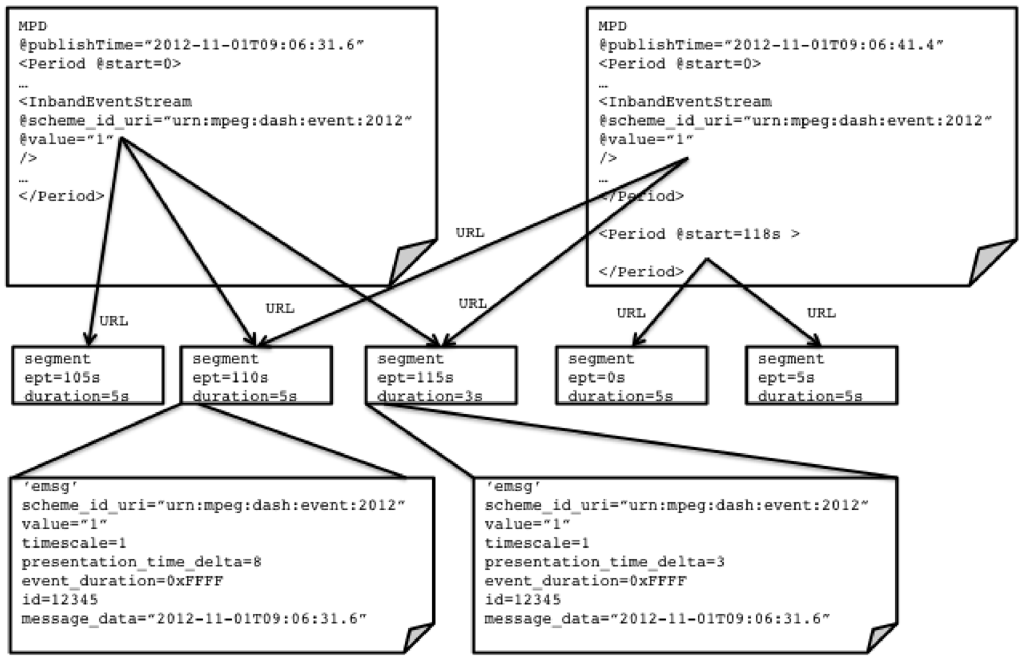

Figure 4 Example for MPD validity expiration to signal new Period

If the `scheme_id_uri` is set to "`urn:mpeg:dash:event:2012`" and the `value` is set to 1, then the fields in the event message box document the following:

- the `message_data` field contains the publish time of an MPD, i.e. the value of the **MPD**`@publishTime`.
- The media presentation time beyond the event time (indicated time by `presentation_time_delta`) is correctly described only by MPDs with publish time greater than indicated value in the `message_data` field.
- the event duration expresses the remaining duration of Media Presentation from the event time. If the event duration is 0, Media Presentation ends at the event time. If 0xFFFF, the media presentation duration is unknown. In the case in which both `presentation_time_delta` and `event_duration` are zero, then the Media Presentation is ended.

This implies that clients attempting to process the Media Presentation at the event time or later are expected to operate on an MPD with a publish time that is later than the indicated publish time in this box.

Note that event boxes in different segments may have identical `id` fields, but different values for `presentation_time_delta` if the earliest presentation time is different across segments.

4.5.2.2. Service Offering

A typical service offering with an Inband event stream is provided in Table 11. In this case the MPD contains information that one or multiple or all Representations contain information that the Representation contains an event message box flow in order to signal MPD validity expirations. The **MPD**`@publishTime` shall be present.

Table 16 – Basic Service Offering with Inband Events

MPD Information	Value
MPD@type	dynamic
MPD@availabilityStartTime	START
MPD@publishTime	PUBTIME1
MPD@minimumUpdatePeriod	MUP
MPD.BaseURL	"http://example.com/"
Period@start	PSTART
InbandEventStream@scheme_id_URI	urn:mpeg:dash:event:2012
InbandEventStream@value	1 or 3
SegmentTemplate@duration	SDURATION

For a service offering based on MPD and segment-based controls, the DASH events shall be used to signal MPD validity expirations.

In this case the following shall apply:

- at least all Representations of all audio Adaptation Sets shall contain an **InbandEvent-Stream** element with `scheme_id_uri = "urn:mpeg:dash:event:2014"` and `@value` either set to 1 or set to 3.
- for each newly published MPD, that includes changes that are not restricted to any of the following (e.g. a new Period):
    - The value of the **MPD**@minimumUpdatePeriod is changed,
    - The value of a **SegmentTimeline.S@**r has changed,
    - A new **SegmentTimeline.S** element is added
    - Changes that do not modify the semantics of the MPD, e.g. data falling out of the timeshift buffer can be removed, changes to service offerings that do not affect the client, etc.

  the following shall be done

    - a new MPD shall be published with a new publish time **MPD**@publishTime
    - an 'emsg' box shall be added to each segment of each Representation that contains an **InbandEventStream** element with
        - `scheme_id_uri = "urn:mpeg:dash:event:2012"`
        - `@value` either set to 1 or set to 3
        - If `@value` set to 1 or 3
            - the value of the **MPD**@publishTime of the previous MPD as the `message_data`

In addition, the following recommendations should be taken into account: All Representations of at least one media type/group contain an **InbandEventStream** element with `scheme_id_uri = "urn:mpeg:dash:event:2012"` and `@value` either set to 1 or set to 3.

4.5.3. Client Requirements and Guidelines

4.5.3.1. Introduction

A DASH client is guided by the information provided in the MPD. An advanced client model is shown in Figure 7. In contrast to the client in section 4.4.3.5, the advanced client requires parsing of segments in order to determine the following information:

- to expand the Segment List, i.e. to generate the Segment Availability Start Time as well as the URL of the next Segment by parsing the Segment Index.
- to update the MPD based on Inband Event Messages using the `'emsg'` box with `scheme_id_uri="urn:mpeg:dash:event:2012"` and `@value` either set to 1 or set to 3.

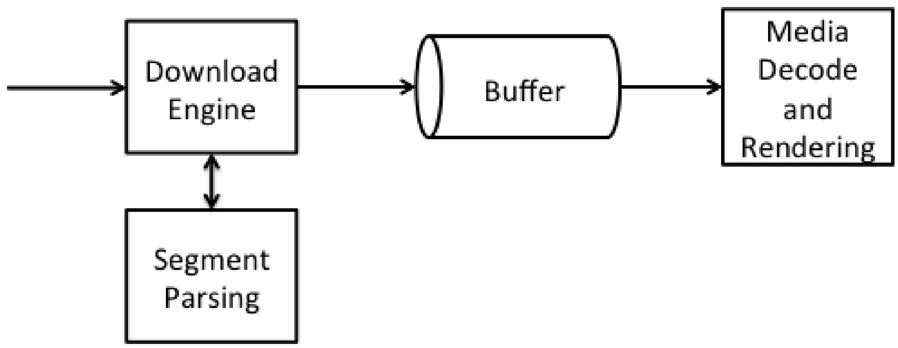

Figure 7 Advanced Client Model

Assumes that the client has access to an MPD and the MPD contains the mandatory parameters in Table 9, i.e., it contains the following information:

- **MPD@**`minimumUpdatePeriod` is set to 0
- **MPD**@publishTime is included and the value is set to PUBTIME
- At least on Representation is present that contains an **InbandEventStream** element with `scheme_id_uri="urn:mpeg:dash:event:2012"` and `@value` either set to 1 or set to 3.
- Either the `@duration` or **SegmentTimeline** for the Representation is present.

In an extension of bullet 7, 8 and 9 in section 4.3.4.1 and section 4.3.4.3, the following example client behaviour may provide a continuous streaming experience to the user as documented in the following.

4.5.3.2. MPD Validity expiration and Updates

The DASH client shall download at least one Representation that contains **InbandEvent-Stream** element with `scheme_id_uri = "urn:mpeg:dash:event:2012"` and `@value` either set to 1 or set to 3. It shall parse the segment at least up to the first `'moof'` box. The DASH client shall parse the segment information and extract the following values:

- `ept` the earliest presentation time of the media segment
- `dur` the media presentation duration of the media segment

If an 'emsg' is detected `scheme_id_uri = "urn:mpeg:dash:event:2012"` and `@value` either set to 1 or set to 3, the DASH client shall parse the segment information and extract the following values:

- `emsg.publish_time` the publish time documented in the message data of the emsg, either directly or from the patch.
- `emsg.ptd` the presentation time delta as documented in the emsg.
- `emsg.ed` the event duration as documented in the emsg

After parsing, the Segment is typically forwarded to the media pipeline if it also used for rendering, but it may either be dumped (if the Representation is only used to access the DASH event, such as muted audio).

If no 'emsg' validity expiration event is included, then

- the current MPD can at least be used up to a media presentation time `ept + dur`

else if an 'emsg' validity expiration event is included, then

- the MPD with publish time equal to `emsg.publish_time` can only be used up to a media presentation time `ept + emsg.ptd`. Note that if `dur > emsg.ptd`, then the Period is terminated at `ept + emsg.ptd`.
- any MPD with publish time greater than `emsg.publish_time` can at least be used up to a media presentation time `ept + emsg.ptd`
- prior to generating a segment request with earliest presentation time greater than `ept + emsg.ptd`, the MPD shall either
  - be refetched and updated by the client.
  - or if `@value=3`, it may be used as included in the message.

NOTE: The DVB DASH profile [42] explicitly forbids downloading a Representation solely to gain access to an Inband Event Stream contained within it. For reference, the relevant part of the DVB DASH specification is section 9.1.6.

4.5.3.3. Extended Segment Information

The DASH client shall download the selected Representation and shall parse the segment at least up to the first `'moof'` box. The DASH client shall parse the segment information and extract the following values:

- `ept` the earliest presentation time of the media segment
  - if the Segment Index is present use the Segments Index
  - if not use the `baseMediaDecodeTime` in `'tfdt'` of the first movie fragment as the earliest presentation time
- `dur` the media presentation duration of the media segment
  - if the Segment Index is present use the Segments Index
  - if not use aggregated sample durations of the first movie fragment as the duration

Using this information, the DASH client should extend the Segment information and, if present the Segment Timeline with the information provided in the Segment. This information can then be used to generate the URL of the next Segment of this Representation. This avoids that the client fetches the MPD, but uses the information of the Segment Timeline. However, in any doubt of the information, for example if a new Adaptation Set is selected, or if Segments or lost, or in case of other operational issues, the DASH client may refetch the MPD in order to obtain the complete information from the MPD.

4.6. Provisioning of Live Content in On-Demand Mode

4.6.1. Scenario

A common scenario for DASH distribution results that a live generated service is also made available for On-Demand offering after the live program is completed. The typical scenario is as follows:

- The Segments as generated for the live service are also used for the On-Demand case. This avoids reformatting and also permits to reuse the Segments that are already cached.
- The MPD is modified to reflect that the content is available as On-Demand now.
- Problems that results from live delivery may be solved, e.g. variable segment durations, or issues of segment unavailability.
- The content may be augmented with ads.
- The content may be trimmed from a longer, e.g. 24/7 stream, at the beginning and/or end.

4.6.2. Content Offering Requirements and Recommendations

In order to provide live content as On-Demand in the above scenario, the following is recommended:

- The same Segments as generated for the live (i.e. **MPD**@type is set to dynamic) distribution are reused also for static distribution (i.e. **MPD**@type is set to static).
- Typically, the Segments also will have the same URL in order to exploit caching advantages.
- An MPD should be generated latest at the end of the live session, but also may be created during an ongoing live session to document a certain window of the program that is offered for On-Demand.
- A new MPD is generated that should contain the following information
  - The **MPD**@type is set to static.
  - The **MPD**@availabilityStartTime should be removed or be maintained from the live MPD since all resources referenced in the MPD are available. If the **MPD**@availabilityStartTime is maintained for a portion of the live program that is offered in the static MPD the **Period**@start value (including the presentation time offset and the start number) and the presentation duration shall be set accordingly. The relationship to the wall-clock time should be maintained by offsetting the **Period**@start without changing the **MPD**@availabilityStartTime.

- As profile, the simple live profile may be used
- The attributes `@timeShiftBufferDepth` and `@minimumUpdatePeriod` shall not be present (in contrast to the live MPD), i.e. it is expected that such attributes are removed. Note that according to ISO/IEC 23009-1, that if present, a client is expected to ignore these attributes for **MPD**`@type` set to `static`.
- The presentation duration is determined through either the `@mediaPresentationDuration` attribute or, if not present, through the sum of the *PeriodStart* and the **Period**`@duration` attribute of the last Period in the MPD.
- Content may be offered in the same Period structure as for live or in a different one.
  - If Periods are continuous, it is preferable to remove the Period structure.
  - If new Periods are added for Ad Insertion, the Periods preferably be added in a way that they are at Segment boundaries.
- Independent whether the `@duration` attribute or the **SegmentTimeline** element was used for the dynamic distribution, the static distribution version may have a **SegmentTimeline** with accurate timing to support seeking and to possibly also signal any gaps in the Segment timeline. To obtain the accurate timeline, the segments may have to be parsed (at least up to the `tfdt`) to extract the duration of each Segment.
- The same templating mode as used in the live service should also be used for static distribution.
- MPD validity expiration events should not be present in the MPD. However, it is not expected that '`emsg`' boxes are removed from Segments.

4.6.3. Client Behavior

For a DASH client, there is basically no difference on whether the content was generated from a live service or the content is provided as On-Demand. However, there are some aspects that may be "left-overs" from a live service distribution that a DASH client should be aware of:

- The Representations may show gaps in the Segment Timeline. Such gaps should be recognized and properly handled. For example a DASH client may find a gap only in one Representation of the content and therefore switches to another Representation that has no gap.
- The DASH client shall ignore any possibly present DASH Event boxes `'emsg'` (e.g., MPD validity expirations) for which no Inband Event Stream is present in the MPD.

4.7. Availability Time Synchronization between Client and Server

4.7.1. Background

According to ISO/IEC 23009-1 [1] and section 4.3, in order to properly access MPDs and Segments that are available on origin servers or get available over time, DASH servers and clients should synchronize their clocks to a globally accurate time standard.

Specifically Segment Availability Times are expected to be wall-clock accurately announced in the MPD and the client needs to have access to the same time base as the MPD generation in order to enable a proper service. In order to ensure this, this section provides server and client requirements to ensure proper operation of a live service.

4.7.2. Service Provider Requirements and Guidelines

If the Media Presentation is dynamic or if the **`MPD`**`@availabilityStartTime` is present then the service shall provide a Media Presentation as follows:

- The segment availability times announced in the MPD should be generated from a device that is synchronized to a globally accurate timing source, preferably using NTP.
- The MPD should contain at least one **`UTCTiming`** element with `@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`
    - `urn:mpeg:dash:utc:ntp:2014`
    - `urn:mpeg:dash:utc:http-head:2014`
    - `urn:mpeg:dash:utc:direct:2014`
- If the MPD does not contain any element **`UTCTiming`** then the segments shall be available latest at the announced segment availability time using a globally accurate timing source.
- If the MPD contains an element **`UTCTiming`** then
    - the announced timing information in the **`UTCTiming`** shall be accessible to the DASH client, and
    - the segments shall be available latest at the announced segment availability time in the MPD for any device that uses one of announced time synchronization methods at the same time.

Despite the latter three technologies may save one or several HTTP transactions, the usage of them should be considered carefully by the MPD author, and rather not be used if the MPD author is not controlling the entire distribution system:

- If `urn:mpeg:dash:utc:ntp:2014` is used, client and server need to implement an NTP client, which may be non-trivial, especially in browser-based clients.
- If `urn:mpeg:dash:utc:http-head:2014` is used, then the server specified in the `@value` attribute of the **`UTCTiming`** element should be the server hosting the DASH segments such that with each request the Date general-header field in the HTTP header (see in RFC 7231 [22], section 7.1.12) can be used by the client to maintain synchroniza-

tion. Also the MPD generator should be aware that caching infrastructures may add inaccuracies to the Date header if the edge caches are not wall-clock synchronized. Therefore, it should not use this method, if they cannot verify that the Date header is set accurately by the edge cache from where each Segment is served. CORS

- If `urn:mpeg:dash:utc:direct:2014` is used, then the MPD generator is expected to write the wall-clock time into the MPD. This basically requires a customized MPD for each request and the MPD should be offered such that it is *not* cached as otherwise the timing is flawed and inaccurate.

Note that in practical deployments segment availability may be an issue due to failures, losses, outages and so on. In this case the Server should use methods as defined in section 4.8 to inform DASH clients about potential issues on making segments available.

A leap second is added to UTC every 18 months on average. A service provider should take into account the considerations in RFC 7164 [50].The MPD time does not track leap seconds. If these occur during a live service they may advance or retard the media against the real time.

4.7.3. Client Requirements and Guidelines

If the Media Presentation is dynamic or if the **MPD**@availabilityStartTime is present then client should do the following:

- If the MPD does not contain any element **UTCTiming** it should acquire an accurate wall-clock time from its system. The anticipated inaccuracy of the timing source should be taken into account when requesting segments close to their segment availability time boundaries.
- If the MPD contains one or several elements **UTCTiming** then the client should at least use one of the announced timing information in the **UTCTiming** to synchronize its clock. The client must not request segments prior to the segment availability start time with reference to any of the chosen **UTCTiming** methods.

  Note: The DVB DASH [42] spec requires support for `http-xsdate` and `http-head` but allows content providers to include others in addition, and allows clients to choose others in preference if they wish. For details, refer to section 4.7 of the DVB DASH specification.

- The client may take into account the accuracy of the timing source as well as any transmission delays if it makes segment requests.
- Clients shall observe any difference between their time zone and the one identified in the MPD, as MPDs may indicate a time which is not in the same timezone as the client.
- If the client observes that segments are not available at their segment availability start time, the client should use the recovery methods defined in section 4.8.
- Clients should not access the **UTCTiming** server more frequently than necessary.

4.8. Robust Operation

4.8.1. Background

In order to support some of the advanced use cases documented in section 2, robust service offerings and clients are relevant. This document lists the relevant ones.

4.8.2. Tools for Robust Operations

4.8.2.1. General Robustness

General Guidelines in ISO/IEC 23009-1 [1] DASH spec in A.7:

- The DASH access client provides a streaming service to the user by issuing HTTP requests for Segments at appropriate times. The DASH access client may also update the MPD by using HTTP requests. In regular operation mode, the server typically responds to such requests with status code 200 OK (for regular GET) or status code 206 Partial Content (for partial GET) and the entity corresponding to the requested resource. Other Successful 2xx or Redirection 3xx status codes may be returned.
- HTTP requests may result in a Client Error 4xx or Server Error 5xx status code. Some guidelines are provided in this subclause as to how an HTTP client may react to such error codes.
- If the DASH access client receives an HTTP client or server error (i.e. messages with 4xx or 5xx error code), the client should respond appropriately (e.g. as indicated in in RFC 7231 [22]) to the error code. In particular, clients should handle redirections (such as 301 and 307) as these may be used as part of normal operation.
- If the DASH access client receives a repeated HTTP error for the request of an MPD, the appropriate response may involve terminating the streaming service.
- If the DASH access client receives an HTTP client error (i.e. messages with 4xx error code) for the request of an Initialization Segment, the Period containing the Initialization Segment may not be available anymore or may not be available yet.
- Similarly, if the DASH access client receives an HTTP client error (i.e. messages with 4xx error code) for the request of a Media Segment, the requested Media Segment may not be available anymore or may not be available yet. In both these case the client should check if the precision of the time synchronization to a globally accurate time standard or to the time offered in the MPD is sufficiently accurate. If the clock is believed accurate, or the error re-occurs after any correction, the client should check for an update of the MPD. . If multiple **`BaseURL`** elements are available, the client may also check for alternative instances of the same content that are hosted on a different server.
- Upon receiving server errors (i.e. messages with 5xx error code), the client should check for an update of the MPD. If multiple **`BaseURL`** elements are available, the client may also check for alternative instances of the same content that are hosted on a different server.

4.8.3. Synchronization Loss of Segmenter

In order to address synchronization loss issues at the segmenter, the following options from the DASH standard should be considered with preference according to the order below:

1. The server is required to always offer a conforming media stream. In case the input stream or encoder is lost, the content author may always add dummy content. This may be done using a separate Period structure and is possible without any modifications of the standard.

2. Usage of the Segment timeline: In general, with every generated segment on the server, the DASH content generator writes the Segment Timeline following the rules in 4.5.2.2. Only if changes are done beyond the ones excluded, an MPD validity expiration is added.

Specifically, when an encoder fails for one or more specific Representations to generate the next Segment, then the DASH content generator may add a segment in the Segment timeline.

3. Early Terminated Periods as included Cor.1 of the second edition of ISO/IEC 23009-1 [4]. Early Terminated Periods may be added that contain both **Period**@start and **Period**@duration. For Early Terminated Periods, the value of the **Period**@duration is the presentation duration in Media Presentation time of the media content represented by the Representations in this Period.The MPD is updated using the @minimumUpdatePeriod, i.e. the timeline is progressing. This permits server to signal that there is an outage of media generation, but that the service is continuing. It is then up to the client to take appropriate actions.

4. Gap filling segments: If the decoder or the playback system supports the ability to overcome losses by providing codec independent gap filling segments (such as a null segment), such an approach may be used as well. No standardized way to support this functionality exists as of today, but it is expected that MPEG will define such functionalities in the near future.

4.8.4. Encoder Clock Drift

In order to support robust offering even under encoder drift circumstances, the segmenter should avoid being synced to the encoder clock. In order to improve robustness, in the case of an MPD-based offering Periods should be added in a period continuous manner. In the case of MPD and segment-based control, the producer reference box should be added to media streams in order for the media pipeline to be aware of such drifts. In this case the client should parse the segment to obtain this information.

4.8.5. Segment Unavailability

To address signaling of segment unavailability between the client and server and to indicate the reason for this, it is recommended to use regular 404s. In addition, unless a UTC Timing has been defined prior in the MPD, the Date-Header specifying the time of the server should be used. In this case, the DASH client, when receiving a 404, knows that if its time is matching the Date Header, then the loss is due to a segment loss.

4.8.6. Swapping across Redundant Tools

To enable swapping across redundant tools doing hot and warm swaps, the following should be considered

1. the content author is offering the service redundant to the client (for example using multiple Base URLs) and the client determines the availability of one or the other. This may be possible under certain circumstances

2. Periods may be inserted at a swap instance in order to provide the new information after swap. If possible, the offering may be period-continuous or period-connected, but the offering may also be non-continuous from a media time perspective.

3. A completely new MPD is sent that removes all information that was available before any only maintains some time continuity. However, this tool is not fully supported yet in any DASH standard and not even considered.

There is a clear preference for the bullets above in their order 1, 2 and 3 as the service continuity is expected to be smoother with higher up in the bullet list. At the same time, it may be the case that the failure and outages are severe and only the third option may be used.

4.8.7. Service Provider Requirements and Guidelines

The requirements and guidelines in subsections 8.2 to 8.6 shall be followed.

4.8.8. Client Requirements and Guidelines

The client shall implement proper methods to deal with service offerings provided in section 8.2 to 8.6.

4.9. Interoperability Aspects

4.9.1. Introduction

In order to provide interoperability based on the tools introduce in this section a restricted set of interoperability points are defined.

4.9.2. Simple Live Operation

4.9.2.1. Definition

The simple live interoperability point permits service offerings with formats defined in the first edition of ISO/IEC 23009-1 [4] as well as in DASH-IF IOPs up to version 2. The DASH client ***is not required to parse media segments for proper operation***, but can rely exclusively on the information in the MPD.

4.9.2.2. Service Requirements and Recommendations

Service offerings conforming to this operation shall follow

- The general requirements and guidelines in section 4.3.3
- the MPD Update requirements and guidelines in section 4.4.3
- the requirements and guidelines for service offering of live content in on-demand mode in section 4.6.2
- the synchronization requirements and guidelines in section 4.7.2
- the robustness requirements and guidelines in section 4.8.7

4.9.2.3. Client Requirements and Recommendations

Clients claiming conformance to this operation shall follow

- The general requirements and guidelines in section 4.3.4
- the MPD Update requirements and guidelines in section 4.4.3.5
- the requirements and guidelines for service offering of live content in on-demand mode in section 4.6.3.
- the synchronization requirements and guidelines in section 4.7.3,
- the robustness requirements and guidelines in section 4.8.8,

4.9.3. Main Live Operation

4.9.3.1. Definition

The main live operation permits service offerings with formats defined in the second edition of ISO/IEC 23009-1 [1]. In this case the DASH client ***may be required to parse media segments for proper operation***.

4.9.3.2. Service Requirements and Recommendations

Service offerings claiming conformance to main live shall follow

- the requirements and guidelines in section 4.3.3
- either
    - the requirements and guidelines in section 4.4.3. Note that in this case no profile identifier needs to be added.
- or
    - the segment-based MPD update requirements and guidelines in section 4.5.2. In this case the profile identifier shall be added.
- the requirements and guidelines for service offering of live content in on-demand mode in section 4.6.2
- the synchronization requirements and guidelines in section 4.7.2
- the robustness requirements and guidelines in section 4.8.7

4.9.3.3. Client Requirements and Recommendations

Clients claiming conformance to main live shall follow

- the requirements and guidelines in section 4.3.4,
- the MPD-update requirements and guidelines in section 4.4.3.5,
- the segment-based MPD update requirements and guidelines in section 4.5.3,
- the requirements and guidelines for service offering of live content in on-demand mode in section 4.6.3.
- the synchronization requirements and guidelines in section 4.7.3,
- the robustness requirements and guidelines in section 4.8.8.

4.10. Trick Mode for Live Services

4.10.1. Introduction

In certain use cases, along with the offering of the main content, a content author also wants to provide a trick mode version primarily of the video Adaptation Set along with the live content that can be used for rewind and fast forward in the time shift buffer of the Media Presentation. In section 3.2.9 signalling is introduced to flag and customize Adaptation Sets for Trick Modes. This clause provides additional service offering requirements and recommendations for trick modes in case of a live service. Typically, a reduced frame rate Representation or an I-frame only version is provided for supporting such trick mode operations.

4.10.2. Service Offering Requirements and Recommendations

If trick mode is to be supported for live services, the trick mode Representations should be offered using the same segment duration as in the main Adaptation Set or each segment duration should aggregate an integer multiple of the segments in the main Adaptation Set. The content author needs to find a balance between the segment duration affecting the amount of requests in fast forward or fast rewind and the availability of trick mode segments at the live edge.

However, longer segment durations for the trick mode Representation delay the Segment availability time of such Segments by the duration of the Segment, i.e. for the live edge the trick mode may not be fully supported. Based on this it is a content author's decision to provide one or more of the following alternatives for trick mode for live services:

- Provide one trick mode Adaptation Set that generates a Segment for every Segment in the main Adaptation Set. Note that if this Adaptation Set is used, it may result in increased amount of HTTP requests when the player does a fast forward or fast rewind.
- Provide one trick mode Adaptation Set that generates a Segment only after several Segments in the main Adaptation Set have been generated and aggregate the trick mode samples in a single Segment of longer duration. This will results that possibly no trick mode samples are available at the live edge
- Provide multiple trick mode Adaptation Sets with different segment durations. If done it is recommended that the `@timeShiftBuffer` for short trick mode segment Adaptation Sets is kept small and in the full timeshift buffer, only trick mode Representations with longer segment durations are maintained. The content author should offer the trick mode Adaptation Sets such that those with longer segment durations can switch to those with shorter duration.
- Provide trick mode Adaptation sets with a single Indexed Media Segment per Period and use Period boundaries with Period connectivity for both, the main Adaptation Set as well as the trick mode Adaptation Set. This means that only for Periods which are not the live Period, trick mode Adaptation Sets are available, or combinations with the above are possible.

4.10.3. Client Implementation Guidelines

If a client wants to access a trick mode Adaptation Set in a live service, it is recommended to minimize the amount of requests to the network, i.e. it should fetch segments with longer segment duration.

4.10.4. Conversion for Live-to-VoD for Trick Mode Adaptation Sets

If the service is converted from live to VoD as described in clause 4.6, it is recommended that trick mode Adaptation Sets are offered with a single Indexed Media Segment per Period.

4.11. Deployment Scenarios

4.11.1. Introduction

This section addresses specifically considered deployment scenarios and provides proposed service configurations based on the technologies introduced in section 4.

4.11.2. Reliable and Consistent-Delay Live Service

4.11.2.1. Scenario Overview

A service provider wants to run a live DASH service according to the below Figure 7. As an example, a generic encoder for a 24/7 linear program or a scheduled live event provides aproduction encoded stream. Such streams typically includ inband events to signal program changes, ad insertion opportunities and other program changes. An example for such signalling are SCTE-35 [54] messages. The stream is then provided to one or more Adaptive Bitrate (ABR) encoders, which transcodes the incoming stream into multiple bitrates and also conditions the stream for segmentation and program changes. These multiple encoders may be used for increased ABR stream density and/are then distributed downstream for redundancy purposes. The resultant streams are received by theDASH generation engines that include: MPD generator, packager and segmenter.Typically the following functions are applied by the MPD packager:

- Segmentation based on in-band information in the streams produced by the ABR encoders
- Encapsulation into ISO BMFF container to generate DASH segments
- Dynamic MPD generation with proper customization options downstream
- Event handling of messages
- Any other other DASH related adaptation

Downstream, the segments may be hosted on a single origin server, or in one or multiple CDNs. The MPD may even be further customized downstream, for example to address specific receivers. Customization may include the removal of certain Adaptation Sets that are not suitable for the capabilities of downstream clients. Specific content may be spliced based on regional services, targeted ad insertion, media blackouts or other information. Events carried from the main encoder may be interpreted and removed by the MPD packager, or they may be carried through for downstream usage. Events may also added as MPD events to the MPD.

In different stages of the encoding and distribution, errors may occur (as indicated by lightning symbols in the diagram), that for itself need to be handled by the MPD Generator and packager, the DASH client, or both of them. The key issue for this section is the ability for the DASH Media Presentation Generator as shown in to generate services that can handle the incoming streams and provide offerings such that DASH clients following DASH-IF IOPs can support.

Hence this section primarily serves to provide guidelines for implementation on MPD Generators and Packagers.

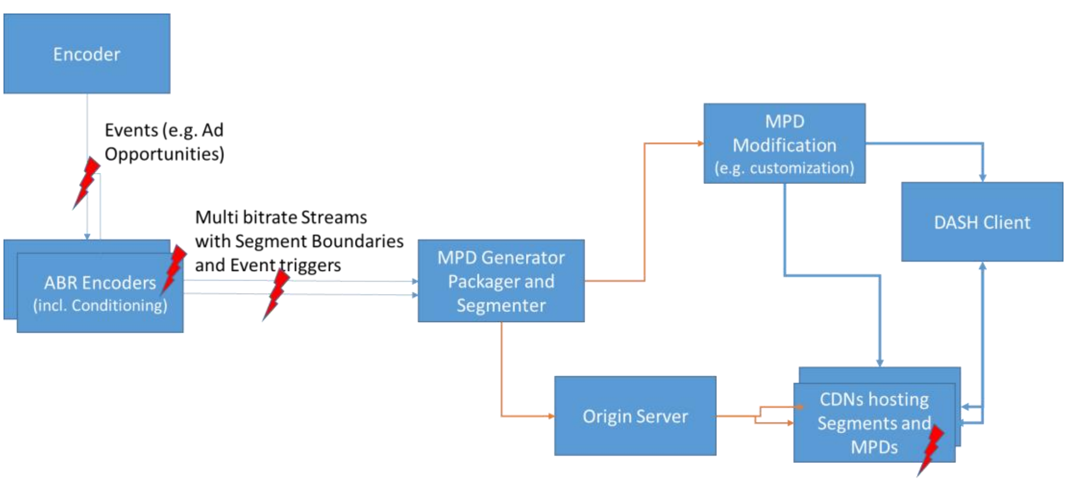

Figure 8 Example Deployment Architecture

More detailed service requirements and recommendations are provided in the following.

4.11.2.2. Service Considerations

The following scenarios are considered in the service setup:

1. The distribution latency should be consistent, typically what is observed for broadcast TV services. This means that the MPD Generator should add minimum delay, and the service should be setup such that the delay between MPD generator and DASH client playout is consistent, and preferably small.
2. Program events may occur for different reasons, for example Program changes, switches from Programs to Advertisements or vice versa, media blackouts or other program changes. Such changes are typically anticipated only on short notice, i.e. within a few seconds. In the following we refer to the time that changes are announced as change lead time. The service should also provide a minimum change lead time, i.e. the smallest time in media time between the change being announced in the stream and the time between the change occurs. Changes may for example include one or more of the following:
    a. Number of source audio languages or formats can change. For example:
        i. Programming with English and Spanish to other content with only English
        ii. Descriptive audio may disappear / reappear
        iii. Programming with 5.1 E-AC-3 and AAC Stereo content to other content with only Stereo AAC
    b. Resolution or format of source video content can change, e.g. HD to/from SD, HDR to/from SDR, etc.
    c. Codecs may change, or at least the profile or level of the codecs
    d. The number of Representations in an Adaptation Set may change
    e. A distribution network may be changed, added or removed.

    As an example, at broadcast origination points if MPEG-2 TS is used, then the Program Map Table (PMT) typically indicates changes such changes. Typically, these changes also result in discontinuities for in the media timeline.
3. The segmentation is determined by the ABR encoders. This encoding may result in occasional slight variations in segment durations during a period (as compared to the last segment in a period) due to encoding optimizations around scene changes near the segment duration point (for example: making a segment slightly shorter or longer to align segment IDR to a scene change).
4. Unanticipated losses and operational failures or outages, possibly happen just for a single encoding (typically at the input of the encoder, but also possibly also downstream packaging). Examples are
    a. An encoder for one or more Representations or the output of an encoder fails for some time and does not produce content.

b. An encoder or the input to the encoder or the output of the encoder fails for a media component/Adaptation Set for some time and do not produce content, e.g. referring to issues as documented in Annex B.4.

c. All encoding or the input to the encoder fails for some time e.g. referring to issues as documented in Annex B.4.

In all cases an MPD can still be written and the MPD is up and running.

Also in the distribution, single Segments may be lost for different reasons and the client typically gets 404.

5. MPD updates should be minimized, whereby MPD updates includes the following aspects for every MPD request

   a. Client sending uplink requests for MPDs

   b. Sending full MPD with every request from the server to the client

   c. Parsing and processing of MPD at the client

   d. Writing a new MPD on the server if the MPD is changed

   All factors are relevant to some extent, but primarily the issues a and b should be minimized.

4.11.3. Relevant DASH-IF IOP Technologies

4.11.3.1. Introduction

This document includes technologies that permit to solve the problems addressed above. We review the available technologies and justify the selection of the technology for the considered scenario. A proposed service configuration is provided in clause 4.11.4.

4.11.3.2. Consistent Latency

The scenario as introduced in clause 4.11.2 does not ask for very low latency, but for consistent latency. In DASH-IF IOP, latency can primarily be controlled by the following means:

- segment duration: the segment duration typically directly impacts the end-to-end latency. Smaller segment sizes provide improved latency and segments of 1-2 seconds may be chosen, if latency is an important aspect. However, too small segments may result in issues, as compression efficiency decreases due to more frequent closed GOPs in the elementary stream. In addition, the number of files/requests to be handled is higher, and finally, with shorter segments, TCP throughput may be such that not the full available capacity on the link can be exploited. Annex B.4 and clause 4.3.3.2.2 provide some guidelines on this.

- If files are available in chunks on the origin, for example due to specific encoding or delivery matters, chunked delivery may be supported. If this feature is offered, then the `@availabilityTimeOffset` attribute may be provided to announce how much earlier than the nominal segment availability the segment can be accessed.

- In order to provide tight synchronization between client and server, and therefore providing the receiver the ability to request the segment at the actual segment availability time, the availability time synchronization as defined in clause 4.7 should be provided and signalled in the MPD. Typically support for `http-xsdate` is sufficient for consistent latency support. Accurate NTP synchronization is recommended, but not required for the MPD packager or the DASH client as long as the time synchronization API is provided.
- It is proposed that a client consistently implements and joins at a segment that is slightly offset (e.g. 4 segments earlier) from the live edge segment. The exact number depends on the distribution system (for example in a fully managed environment, the offset may be smaller in contrast to best effort networks). The MPD author may support consistency by providing a suggested presentation delay in the service offering. For details on joining at the live edge, please refer to clause 4.3.4.4.2.

4.11.3.3. Unanticipated New Periods

To avoid that the clients take future segment existence for granted even if a sudden change on the service offering is necessary, the MPD service provider must set to the **MPD**`@minimumUpdatePeriod` to a low value. All Segments with availability start time less than the sum of the request time and the value of the **MPD**`@minimumUpdatePeriod` will eventually get available at the advertised position at their computed segment availability start time.

In the most conservative case, the MPD author sets the **MPD**`@minimumUpdatePeriod` to 0. Then only Segments with availability start time less than the request time are available, i.e. no promise for future segments is provided. The DASH client is forced to revalidate the MPD prior to any new Segment request. For this purpose, basically two options exists:

- Option 1) Client revalidates MPD with every Segment request according to clause 4.4.4, preferably using a conditional GET in order to avoid unnecessary downlink traffic and processing in the client.
- Option 2) Client relies on MPD validity expiration events in event messages, if content provider announces those in the MPD and by this, it can revalidate.

Note that the two methods are not mutually exclusive. More details are discussed further below.

In case of option 1 using MPD level validation, with every generated segment on the server, the DASH content generator checks the validity of the MPD offering. If still valid, no changes to the MPD are done. Only if changes are done that are no longer valid, a new MPD is written.

4.11.3.4. Segment Duration Variations

Variable segment durations impact the accuracy of the MPD times of the Segments. MPD times are used for the computation of the segment availability time. With variable segment durations, the segment availability times vary and can impact the DASH-IF IOPs basically provide to options to deal with variable segment durations

- Option 1)

- Signalling of constant segment duration using `@duration`, permitting a variation of +/- 50% of the segment duration. According to clause 3.2.7.1, for each media segment in each Representation the MPD start time of the segment should approximately be *EPT* - *PTO*. Specifically, the MPD start time shall be in the range of *EPT* - *PTO* - 0.5**DUR* and *EPT* - *PTO* + 0.5**DUR* according to the requirement stated above.

  Note that the encoder should provide segments of a virtual segmentation that adheres to this rule. However, there may be reasons that the encoder does break this rule occasionally.

- If the DASH packager receives a segment stream such that the drift can no longer be compensated, then a new Period should be added, that adjusts the parameters for the segment availability computation, but also signals that the Period is continuous as defined in 4.3.3.3.2. Note that this is done for all Representations in the MPD and a change of the MPD is happening, i.e. this needs to be anncouned. However, no segment parsing is necessary for the client.

- Option 2) Following the rules in 4.5.2.2 and using the Segment Timeline to accurately signal the different segment durations. If the segment duration changes, then the `@r` attribute of the last **`S`** element in the Segment timeline is terminated and a new S element is added to the MPD with the new segment duration. Note that this results in a change of the MPD. The client should determine such changes independent of MPD updates by detailed segment parsing to obtain the earliest presentation time of the segment and the segment duration.

4.11.3.5. Losses and Operational Failures

One of the most complex aspects are occasional operational issues, such as losses, outages, failovers of input streams, encoders, packagers and distribution links. Section 4.8 provides detailed overview on available tools that should be used by network service offering and clients in order to deal with operational issues. Several types of losses may occur as shown in Figure 9:

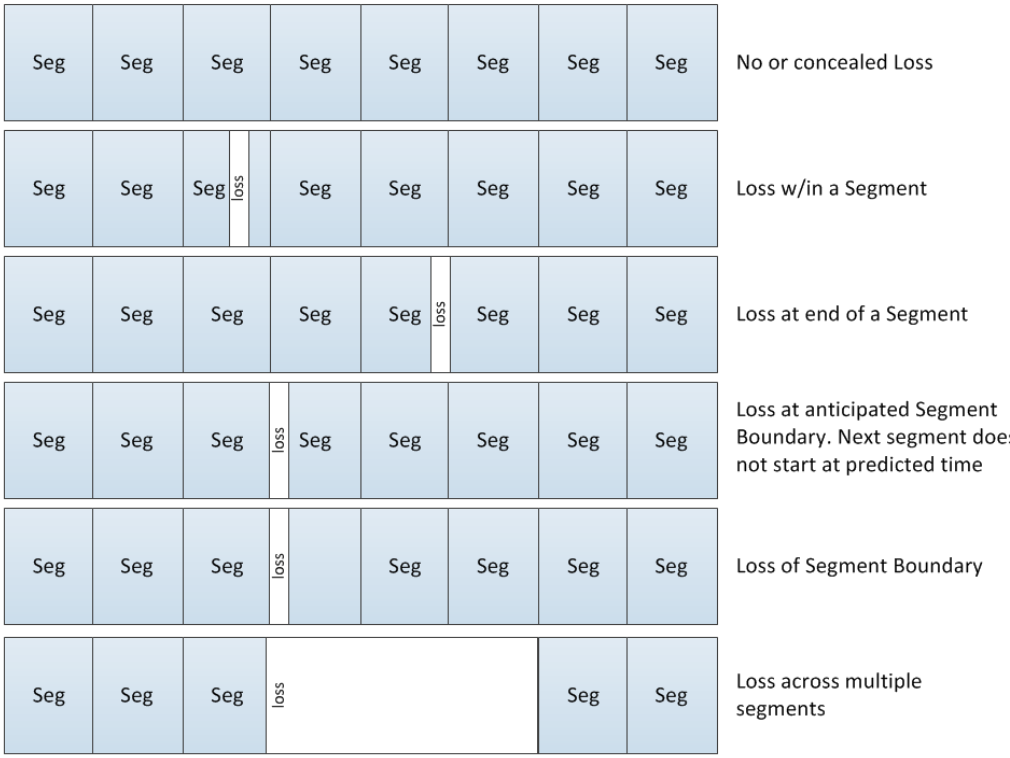

Figure 9 Loss scenarios

Losses may occur in the middle of a Segment, at the end of a Segment, at the start of a new Segment. At the elementary stream level, losses may be within a compressed access unit (AU), producing a syntactically corrupt bitstream, or may be the result of the ABR encoder simply not encoding a source frame in which case the duration of the prior AU is extended producing a conforming bitstreams. Losses may impact an entire Segment or may just impact a part of the Segment. Typically, service oriented losses will occur until the next Random access point, i.e. a loss is to be signaled from the start of the lost sample up to the next random access point, typically coinciding with the start of a new Segment.

In order to deal with this, the MPD packager basically has the following options that are not mutually exclusive:

- Option 1: Operation with Main live profile expecting main profile clients. In this case the Segment Timeline is used to signal gaps in the media. For details see below.
- Option 2: Operation with simple live profile expecting simple profile clients. In this case the early terminated Period is used to signal gaps in the media. For details see below.
- Option 3: Gap filling segments. Note that Gap filling segments are not part of the IOP recommendations as the DASH client would be unaware that such gaps are happening. The DASH client can not take proper measures, such as switching to a Representation that is not under loss and does provides the actual content. Therefore, gap filling segments should be used by the content author carefully taking this into account.

Note: At the time of writing we are aware that MPEG is addressing the issue of gap filling segments. It is expected that at foreseeable time an update on this issue will be provided.

In addition to the above, the content provider may offer to provide the same content on different Base URLs. In this case, the temporary non-availability may be signaled as well through the MPD.

4.11.3.6. Minimizing MPD Updates

MPD updates, the frequency of MPD updates and the actions included in MPD updates are different ones, and their effects may have different impacts on deployments. To avoid confusion on the generally overloaded term, some more details are discussed in the following section. In non-DASH adaptive streaming solutions, MPD updates result in the following additional processing and delivery overhead:

1. The client sends an uplink requests for the MPD. At least from a CDN perspective, this is issue is considered less critical, typically the bounds of operation are reached by throughout, not by the number of requests.

2. The server needs to send a full MPD with every request, which for itself causes overhead from all the way of the origin server to the client. This is in particular relevant if the manifest contains a list of URLs, and some timeshift buffer is maintained.

3. Yet another aspect is the regular parsing and processing of the manifest in the client. Whereas the processing is likely less of a burden, the consistency across two parsing instances is relevant and requires to keep state.

4. MPD updates may also result in writing a new MPD on the server. This may be less problematic for certain cases, especially for unicast, but it results in significant overhead if DASH formats are used for broadcast.

DASH-IF IOP provides different means to avoid one or the more of the above issues. Assuming that the **MPD**`@minimumUpdatePeriod` is set to a low value for reasons documented above, then issues mentioned above can be addressed by the following means in DASH-IF IOP

1. Client Requests: can be avoided by signalling inband that an MPD is has expired. The most obvious tool is the use of Inband Events with MPD expiry. However, this requires inband events being added during packaging.

2. Sending Full MPD: Instead of requesting the full MPD, the client can support this operation by issuing a conditional GET. If the MPD has not changed, no MPD needs to be sent and the downlink rate is small. However, this requires the usage of `@duration` or **`SegmentTimeline`** with `@r=-1`.

3. MPD Parsing and Processing: This can be avoided by using either of the solutions documented above.

4. MPD writing on server: This goes hand-in-hand with 2, i.e. the usage of `@duration` or **`SegmentTimeline`** with `@r=-1`.

Generally, DASH-IF IOP provide several tools to address different aspects of minimizing MPD updates. Based on the deployment scenario, the appropriate tools should be used. However, it is preferable that DASH clients support different tools in order to provide choices for the service offering.

4.11.4. Proposed Service Configuration and Generation of the MPD and Segments based on a “Segment Stream”

4.11.4.1. Introduction and Assumptions

The core concept is the availability of a segment stream at the input to a packager. The segment stream may be made available as individual segments or as boundary markers in a continuous stream. In addition, the stream may contain information that is relevant for the packager, such as program changes. The segment stream determines for each segment the earliest presentation time, the presentation duration, as well as boundaries in program offerings.

Furthermore, it is assumed that multiple bitrates may exist that are switchable. In the following we focus on one segment stream, but assume that in the general case multiple bitrates are available and the encoding and segment streams are generated such that they can be switched.

The high-level assumptions for the service are summarized in 4.11.2. Based on these assumptions, a more detailed model is provided.

- A segment stream is provided for each Representation. The segmentation is the same for Representations that are included in one Adaptation Set. Each segment *i* has assigned a duration *d*[*i*] and an earliest presentation time *ept*[*i*]. In addition, the segment stream has a nominal segment duration *d0* that the ABR encoders attempts to maintain. However, variation may occur for different reasons, documented above.
- Losses may occur in the segment stream, spanning a part of a segment, multiple segments, a full segment and so on. The loss may be in one Representation or in multiple Representations at the same time (see above for more discussions).
- The latency of the time that the segment is made available to the DASH packager and that it is offered as an available segment in the MPD should be small, i.e. the segment availability time should be shortly after the time when the full segment is received in the DASH packager. Any permitted delay by the MPD Packager can be view as additive to change lead time and may therefore improve efficiency and robustness, but may at the same time increase the end-to-end latency.
- Changes in the program setup may occur, that signal changes as discussed in 4.11.2. A change is possibly announced with a time referred to as change lead time. Note that signal changes such as SCTE-35 only indicate where a change may occur, it does not indicate what type of change will occur.

The different scenarios are summarized in Figure xxx.

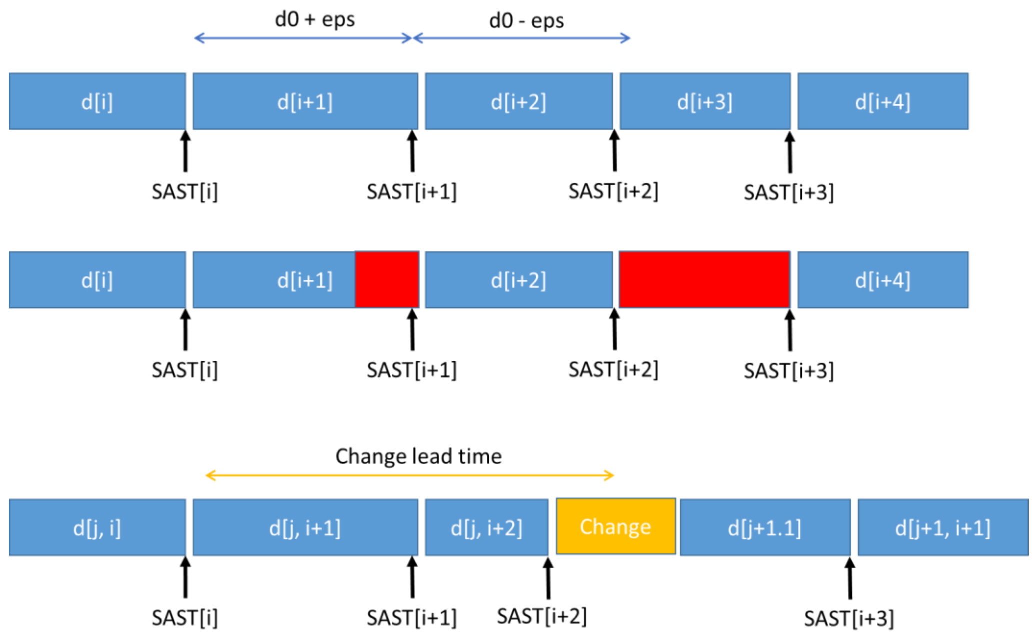

Figure 10: Different properties of a segment stream

Based on the discussions in 4.11.2, proposed service configuration for such a service are proposed. The service configuration differentiates two deployment scenarios:

1) Clients implementing the simple live client, i.e. no emsg support and no segment parsing is implemented.

2) Clients implementing the main client, i.e. emsg is supported and segment parsing is implemented.

In the following, reference is made to technologies in section 4.11.3.

4.11.4.2. Service Configuration for Simple Live

Assuming that the input stream is a segment stream with the properties documented above is received by the DASH packager.

The DASH packager may operate as follows:

- The `@minimumUpdatePeriod` is set to a value that is equal or smaller than the change lead time provided by the segment stream.
- The `@timescale` of the Adaptation Set is set to the timescale of the included media
- The `@duration` attribute is set such that the nominal duration d0 is documented in the MPD for this Adaptation Set.
- `$Number$` is used of segment templating.
- With incoming segments of the segment stream, a new segment is generated 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 next 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 `@startNumber` is set to the first segment in the new Period.
    - The Adaptation Sets are continued by providing Period continuity signallng with each Adaptation Set.
- when an encoder fails for one or more specific Representations to generate the next segment, then the DASH content generator
  - terminates the Segment with the last sample in the segment, (which is possibly corrupted)
  - generates a new MPD as follows:
    - The `@minimumUpdatePeriod` is set to `0`.
    - If all or at least many Representations fail, the Period`@duration` is set to the value of the media time in the Period that is still available.
    - If only a subset of the Representations fail, the `@presentationDuration` for the last segment is set to the value of the last presentation time in the Representation that is still available.
    - By doing so, the content provider basically informs the DASH client that for the duration of the Segment as announced, no media is available. The DASH client revalidates this after every Segment duration. The MPD is not changed on the server until either the decoder resumes or the Media Presentation is terminated.
    - If the `@minimumUpdatePeriod` is long, then the client may request non-existent segments, which itself may then trigger that the DASH client revalidates the MPD. If the DASH client has the possibility, it should add the '`lmsg`' brand as a compatibility brand to the last generated segment. In addition, when the segment is distributed over HTTP, the HTTP header should signal the content type of the segment including the compatibility brand '`lmsg`'. If the DASH client can identify this, it is expected to refetch the MDP and may by this means observe the early terminated Period or Representations.

- Only after the encoder resumes, a new MPD is written as follows:
  - A new Period is provided with **Period**@start according to the value of the new Period. The @presentationTimeoffset of the Representation of the Period shall match the the earliest presentation time of the newly generated Segment. If appropriate, Period connectivity should be signaled.
  - The @minimumUpdatePeriod is set again to the minimum change lead time.

- 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

4.11.4.3. Service Configuration for Main Live

Assuming that the input stream is a segment stream with the properties documented above is received by the DASH packager.

The DASH packager may operate as follows:

- The @minimumUpdatePeriod is set to 0.
- The @timescale of the Adaptation Set is set to the timescale of the included media
- The segment timeline is used. Addressing may used: $Number$ or $Time$.
- The MPD is assigned an **MPD**@publishTime
- With incoming segments of the segment stream, following the rules in 4.5.2.2 the DASH Packager uses the Segment Timeline to accurately signal the different segment durations. If the segment duration changes, then the @r attribute of the last **S** element in the Segment timeline is terminated and a new S element is added to the MPD with the new segment duration. The values @t and @d need to be set correctly:
  - @r of the last segment element may be set to -1. In this case a new MPD is only written if the segment duration changes
  - @r of the last segment element may be set to the actual published number of segments. In this case a new MPD is written for each new segment
- Whenever a new MPD is written, the **MPD**@publishTime is updated.

- when an encoder fails for one or more specific Representations to generate the next segment, then the DASH packager
    - terminates the Segment with the last sample in the segment (may be corrupt)
    - adds `emsg` to this last generated segment. The MPD validity expiration is set to the duration of the current segment or smaller. This `emsg` may be added to all Representation that have observed this failure, to all Representations in the Adaptation Set or to all Representations in the MPD. The content author should be aware that if the emsg is not signaled with all Representations, then there exist cases that a switch *to* the erroneous Representation causes a request to a non-existing Segment. That loss would be signaled in the MPD, but the client is not aware that an update of the MPD is necessary.
    - The `emsg` shall be added to all Representations that announce that they carry the message as an inband stream.
    - The MPD is updated on the server such that the last generated segment is documented in the Segment timeline and no new **`S`** element is added to the timeline.
    - Only after the Representation(s) under loss resumes, a new **`S`** element is written with **`S`**`@t` matching the earliest presentation time of the newly generated Segment. The DASH client with it next update will resume and possibly take into account again this Representation.
    - If the encoder does not resume for a specific Representation over a longer time, it is recommended to terminate this Period and remove this Representation at least temporarily until the encoder resumes again. Period continuity should be signaled.
- when the program change occurs
    - adds `emsg` to this last generated segment. The MPD validity expiration is set to the duration of the current segment or smaller. This `emsg` shall be added to all Representations that announce the Inband Event stream for the MPD validity expiration.
    - Write a new MPD with all the parameters
- Whenever a new MPD is written, the **`MPD`**`@publishTime` is updated.

4.11.5. Client Support Considerations

4.11.5.1. Introduction

Generally the client should support the rules in this section for the specific clients.

4.11.5.2. Client Requirements for Simple Live

The client shall follow the details in clause 4.3.4 and 4.4.4. In addition, the DASH client is expected to handle any losses signalled through early termined Periods.

4.11.5.3. Client Requirements for Main Live

The client shall follow the details in clause 4.3.4 and 4.5.3. In addition, the DASH client is expected to handle any losses signalled through gaps in the segment timeline.

The DASH client having received an MPD that signals gaps is expected to either look for alternative Representations that are not affected by the loss, or if not possible, do some appropriate error concealment. The DASH client also should go back regularly to check for MPD updates whether the Representation gets available again.

5. Ad Insertion in DASH

5.1. Introduction

5.1.1. General

This section provides recommendations for implementing ad insertion in DASH. Specifically, it defines the reference architecture and interoperability points for a DASH-based ad insertion solution.

The baseline reference architecture addresses both server-based and app-based scenarios. The former approach is what is typically used for Apple HLS, while the latter is typically used with Microsoft SmoothStreaming and Adobe HDS.

5.1.2. Definitions

The following definitions are used in this section:

Ad Break: A location or point in time where one or more ads may be scheduled for delivery; same as *avail* and *placement opportunity*.

Ad Decision Service: functional entity that decides which ad(s) will be shown to the user. It interfaces deployment-specific and are out of scope for this document.

Ad Management Module: logical service that, given cue data, communicates with the ad decision service and determines which advertisement content (if at all) should be presented during the ad break described in the cue data.

Cue: indication of time and parameters of the upcoming ad break. Note that cues can indicate a pending switch to an ad break, pending switch to the next ad within an ad break, and pending switch from an ad break to the main content.

CDN node: functional entity returning a segment on request from DASH client. There are no assumptions on location of the node.

Packager: functional entity that processes conditioned content and produces media segments suitable for consumption by a DASH client. This entity is also known as fragmenter, encapsulater, or segmenter. Packager does not communicate directly with the origin server – its output is written to the origin server's storage.

Origin: functional entity that contains all media segments indicated in the MPD, and is the fallback if CDN nodes are unable to provide a cached version of the segment on client request.

Splice Point: point in media content where its stream may be switched to the stream of another content, e.g. to an ad.

MPD Generator: functional entity returning an MPD on request from DASH client. It may be generating an MPD on the fly or returning a cached one.

XLink resolver: functional entity which returns one or more remote elements on request from DASH client.

5.1.3. DASH Concepts

5.1.3.1. Introduction

DASH ad insertion relies on several DASH tools defined in the second edition of ISO/IEC 23009-1 [4], which are introduced in this section. The correspondence between these tools and ad insertion concepts are explained below.

5.1.3.2. Remote Elements

Remote elements are elements that are not fully contained in the MPD document but are referenced in the MPD with an HTTP-URL using a simplified profile of XLink.

A remote element has two attributes, `@xlink:href` and `@xlink:actuate`. `@xlink:href` contains the URL for the complete element, while `@xlink:actuate` specifies the resolution model. The value "`onLoad`" requires immediate resolution at MPD parse time, while "`onRequest`" allows deferred resolution at a time when an XML parser accesses the remote element. In this text we assume deferred resolution of remote elements, unless explicitly stated otherwise. While there is no explicit timing model for earliest time when deferred resolution can occur, the specification strongly suggests it should be close to the expected playout time of the corresponding Period. A reasonable approach is to choose the resolution at the nominal download time of the Segment.

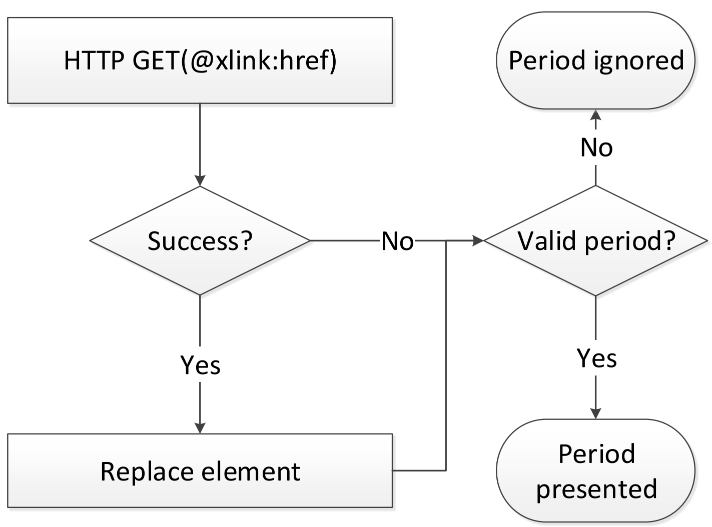

Figure 11: XLink resolution

Resolution (a.k.a. dereferencing) consists of two steps. Firstly, a DASH client issues an HTTP GET request to the URL contained in the `@xlink:href`, attribute of the *in-MPD element*, and the XLink resolver responds with a *remote element entity* in the response content. In case of error response or syntactically invalid remote element entity, the `@xlink:href` and `@xlink:actuate` attributes the client shall remove the *in-MPD element*.

If the value of the `@xlink:href` attribute is `urn:mpeg:dash:resolve-to-zero:2013`, HTTP GET request is not issued, and the in-MPD element shall be removed from the MPD. This special case is used when a remote element can be accessed (and resolved) only once during the time at which a given version of MPD is valid.

If a syntactically valid remote element entity was received, the DASH client will replace in-MPD element with remote period entity.

Once a remote element entity is resolved into a fully specified element, it may contain an `@xlink:href` attribute with `@xlink:actuate` set to `'onRequest'`, which contains a new XLink URL allowing repeated resolution.

Note that the only information passed from the DASH client to the XLink resolver is encoded within the URL. Hence there may be a need to incorporate parameters into it, such as splice time (i.e., *PeriodStart* for the remote period) or cue message.

> Note: In ISO/IEC 23009-1:2014/Cor.3 it is clarified that if multiple top-level remote elements are included, the remote element entity is not a valid XML document.

5.1.3.3. Periods

5.1.3.3.1. Timing

Periods are time-delimited parts of a DASH Media Presentation. The value of *PeriodStart* can be explicitly stated using the **Period@**`start` attribute or indirectly computed using **Period**`@duration` of the previous Periods.

Precise period duration of period *i* is given by *PeriodStart(i+1) – PeriodStart(i)*. This can accommodate the case where media duration of period *i* is slightly longer than the period itself, in which case a client will schedule the start of media presentation for period *i+1* at time *PeriodStart(i+1)*.

Representation`@presentationTimeOffset` specifies the value of the presentation time at *PeriodStart(i)* .

5.1.3.3.2. Segment Availability

In case of dynamic MPDs, Period-level **BaseURL**`@availabilityTimeOffset` allow earlier availability start times. A shorthand notation `@availabilityTimeOffset="INF"` at a Period-level **BaseURL** indicates that the segments within this period are available at least as long as the current MPD is valid. This is the case with stored ad content. Note that DASH also allows specification of `@availabilityTimeOffset` at Adaptation Set and Representation level.

5.1.3.3.3. Seamless transition

The DASH specification says nothing about Period transitions – i.e., there are no guarantees for seamless continuation of playout across the period boundaries. Content conditioning and receiver capability requirements should be defined for applications relying on this functionality. However, Period continuity or connectivity should be used and signaled as defined in section 3.2.12 and ISO/IEC 23009-1:2014/Amd.3 [4].

5.1.3.3.4. Period labeling

Period-level **AssetIdentifier** descriptors identify the asset to which a given Period belongs. Beyond identification, this can be used for implementation of client functionality that depends on distinguishing between ads and main content (e.g. progress bar and random access).

5.1.3.4. DASH events

DASH events are messages having type, timing and optional payload. They can appear either in MPD (as period-level event stream) or inband, as ISO-BMFF boxes of type `` `emsg` ``. The `` `emsg` `` boxes shall be placed at the very beginning of the Segment, i.e. prior to any media data, so that DASH client needs a minimal amount of parsing to detect them.

DASH defines three events that are processed directly by a DASH client: MPD Validity Expiration, MPD Patch and MPD Update. All signal to the client that the MPD needs to be updated – by providing the publish time of the MPD that should be used, by providing an XML patch that can be applied to the client's in-memory representation of MPD, or by providing a complete new MPD. For details please see section 4.5.

User-defined events are also possible. The DASH client does not deal with them directly – they are passed to an application, or discarded if there is no application willing or registered to process these events. A possible client API would allow an application to register callbacks for specific event types. Such callback will be triggered when the DASH client parses the `` `emsg` `` box in a Segment, or when it parses the **`Event`** element in the MPD.

In the ad insertion context, user-defined events can be used to signal information, such as cue messages (e.g. SCTE 35 [54])

5.1.3.5. MPD Updates

If **`MPD`**`@minimumUpdatePeriod` is present, the MPD can be periodically updated. These updates can be *synchronous*, in which case their frequency is limited by **`MPD`**`@minimumUpdatePeriod.` In case of the main live profiles MPD updates may be triggered by DASH events. Fir details refer to section 4.5.

When new period containing stored ads is inserted into a linear program, and there is a need to unexpectedly alter this period the inserted media will not carry the `` `emsg` `` boxes – these will need to be inserted on-the-fly by proxies. In this case use of synchronous MPD updates may prove simpler.

`MPD``@publishTime` provides versioning functionality: MPD with later publication times include all information that was included all MPDs with earlier publication times.

5.1.3.6. Session information

In order to allow fine-grain targeting and personalization, the identity of the client/viewer, should be known i.e. maintain a notion of a session.

HTTP is a stateless protocol, however state can be preserved by the client and communicated to the server.

The simplest way of achieving this is use of cookies. According to RFC 6265 [41], cookies set via 2xx, 4xx, and 5xx responses must be processed and have explicit timing and security model.

5.1.3.7. Tracking and reporting

The simplest tracking mechanism is server-side logging of HTTP GET requests. Knowing request times and correspondence of segment names to content constitutes an indication that a certain part of the content was requested. If MPDs (or remote element entities) are generated on the fly and identity of the requester is known, it is possible to provide more precise logging. Unfortunately this is a non-trivial operation, as same user may be requesting parts of content from different CDN nodes (or even different CDNs), hence log aggregation and processing will be needed.

Another approach is communicating with existing tracking server infrastructure using existing external standards. An IAB VAST-based implementation is shown in section 5.3.3.7.

DASH Callback events are defined in ISO/IEC 23009-1:2014 AMD3 [4], are a simple native implementation of time-based impression reporting (e.g., quartiles). A callback event is a promise by the DASH client to issue an HTTP GET request to a provided URL at a given offset from *PeriodStart*. The body of HTTP response is ignored. Callback events can be both, MPD and inband events.

5.2. Architectures

The possible architectures can be classified based on the location of component that communicates with the ad decision service: a *server-based* approach assumes a generic DASH client and all communication with ad decision services done at the server side (even if this communication is triggered by a client request for a segment, remote element, or an MPD. The *app-based* approach assumes an application running on the end device and controlling one or more generic DASH clients.

Yet another classification dimension is amount of media engines needed for a presentation – i.e., whether parallel decoding needs to be done to allow seamless transition between the main and the inserted content, or content is conditioned well enough to make such transition possible with a single decoder.

Workflows can be roughly classified into *linear* and *elastic*. Linear workflows (e.g., live feed from an event) has ad breaks of known durations which have to be taken: main content will only resume after the end of the break and the programmer / operator needs to fill them with some inserted content. Elastic workflows assume that the duration of an ad break at a given cue location not fixed, thus the effective break length can vary (and can be zero if a break is not taken).

5.3. Server-based Architecture

5.3.1. Introduction

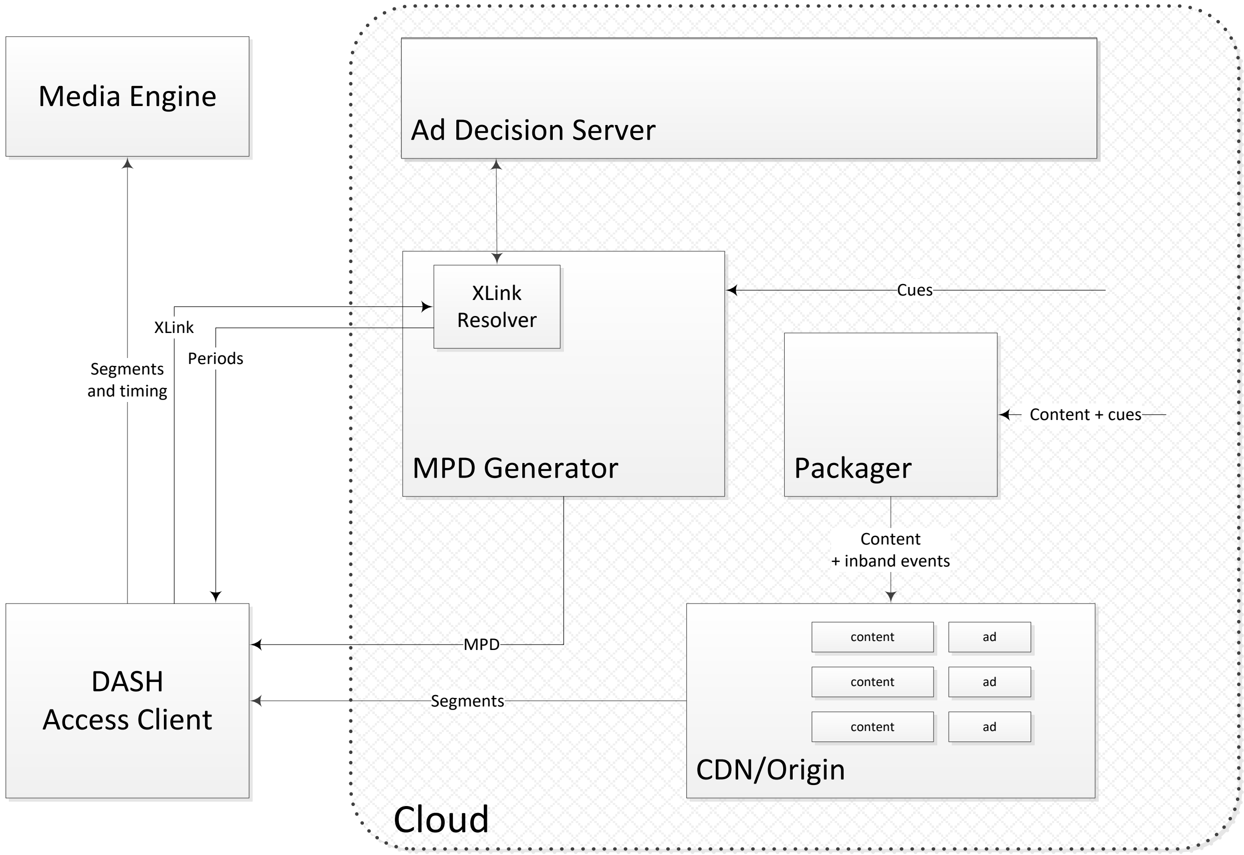

Figure 12: Server-based architecture

In the server-based model, all ad-related information is expressed via MPD and segments, and ad decisions are triggered by client requests for MPDs and for resources described in them (Segments, remote periods).

The server-based model is inherently MPD-centric – all data needed to trigger ad decision is concentrated in the MPD. In case where ad break location (i.e., its start time) is unknown at the MPD generation time, it is necessary to rely on MPD update functionality. The two possible ways of achieving these are described in 5.1.3.5.

In the live case, packager receives feed containing inband cues, such as MPEG-2 TS with SCTE 35 cue messages [54]. The packager ingests content segments into the CDN. In the on demand case, cues can be provided out of band.

Ad management is located at the server side (i.e., in the cloud), thus all manifest and content conditioning is done at the server side.

5.3.2. Mapping to DASH

5.3.2.1. Period elements

5.3.2.1.1. General

A single ad is expressed as a single `Period` element.

Periods with content that is expected to be interrupted as a result of ad insertion should contain explicit start times (**Period**@start), rather than durations. This allows insertion of new periods without modifying the existing periods. If a period has media duration longer then the distance between the start of this period and the start of next period, use of start times implies that a client will start the playout of the next period at the time stated in the MPD, rather than after finishing the playout of the last segment.

An upcoming ad break is expressed as Period element(s), possibly remote.

5.3.2.1.2. Remote Period elements.

Remote Periods are resolved on demand into one or more than one Period elements. It is possible to embed parameters from the cue message into the XLink URL of the corresponding remote period, in order to have them passed to the ad decision system via XLink resolver at resolution time.

In an elastic workflow, when an ad break is not taken, the remote period will be resolved into a period with zero duration. This period element will contain no adaptation sets.

If a just-in-time remote Period dereferencing is required by use of @xlink:actuate="onRequest", MPD update containing a remote period should be triggered close enough to the intended splice time. This can be achieved using MPD Validity events and full-fledged MPD update, or using MPD Patch and MPD Update events (see sec. 5.1.3.5 and 5.1.3.4). However, due to security reasons MPD Patch and MPD Update events should only be used with great care.

In case of **Period**@xlink:actuate="onRequest", MPD update and XLink resolution should be done sufficiently early to ensure that there are no artefacts due to insufficient time given to download the inserted content. Care needs to be taken so that the client is given a sufficient amount of time to (a) request and receive MPD update, and (b) dereference the upcoming remote period.

NOTE: It may be operationally simpler to avoid use of **Period**@xlink:actuate="onRequest", dereferencing in case of live content.

5.3.2.1.3. Timing and dereferencing

The only interface between DASH client and the XLink resolver is the XLink URL (i.e., the **Period**@xlink:href attribute).After resolution, the complete remote Period element is replaced with Period element(s) from the remote entity (body of HTTP response coming from XLink resolver). This means that the XLink resolver is (in the general case) unaware of the exact start time of the ad period.

In case of linear content, start of the ad period is only known a short time before the playback. The recommended implementation is to update the MPD at the moment the start of the ad period is known to the MPD generator.

The simplest approach for maintaining time consistency across dereferencing is to have the MPD update adding a **Period**@duration attribute to the latest (i.e., the currently playing) main content period. This means that the MPD resolver needs to include the **Period**@duration attribute into each of the Period elements returned in the remote entity. The downside of this approach is that the DASH client needs to be able to update the currently playing period.

An alternative approach is to embed the desired value of **Period**@start of the first period of the remote entity in the XLink URL (e.g., using URL query parameters). This approach is described in clause 5.3.5. The downside of this alternative approach is that the DASH specification does not constrain XLink URLs in any way, hence the XLink resolver needs to be aware of this URL query parameter interface defined in clause 5.3.5.

5.3.2.2. Asset Identifiers

`AssetIdentifier` descriptors identify the asset to which a Period belongs. This can be used for implementation of client functionality that depends on distinguishing between ads and main content (e.g. progress bar).

Periods with same **`AssetIdentifier`** should have identical Adaptation Sets, Initialization Segments and same DRM information (i.e., DRM systems, licenses). This allows reuse of at least some initialization data across periods of the same asset, and ensures seamless continuation of playback if inserted periods have zero duration. Period continuity or connectivity should be signaled, if the content obeys the rules.

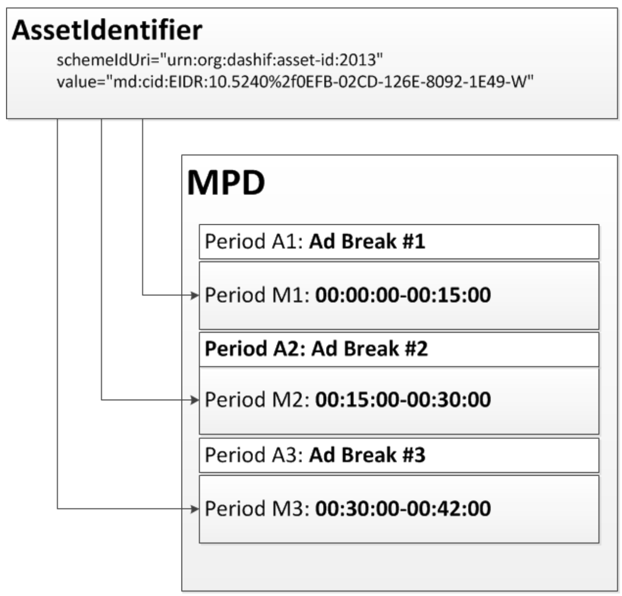

Figure 13 Using an Asset Identifier

5.3.2.3. MPD updates

MPD updates are used to implement dynamic behavior. An updated MPD may have additional (possibly – remote) periods. Hence, MPD update should be triggered by the arrival of the first cue message for an upcoming ad break. Ad breaks can also be canceled prior to their start, and such cancellation will also trigger an MPD update.

Frequent regular MPD updates are sufficient for implementing dynamic ad insertion. Unfortunately they create an overhead of unnecessary MPD traffic – ad breaks are rare events, while MPD updates need to be frequent enough if a cue message is expected to arrive only several seconds before the splice point. Use of HTTP conditional GET requests (i.e., allowing the server to respond with "304 Not Modified" if MPD is unchanged) is helpful in reducing this overhead, but asynchronous MPD updates avoid this overhead entirely.

DASH events with scheme "`urn:mpeg:dash:event:2013`" are used to trigger asynchronous MPD updates.

The simple mapping of live inband cues in live content into DASH events is translating a single cue into an MPD Validity expiration event (which will cause an MPD update prior to the splice time). MPD Validity expiration events need to be sent early enough to allow the client request a new MPD, resolve XLink (which may entail communication between the resolver and ADS), and, finally, download the first segment of the upcoming ad in time to prevent disruption of service at the splice point.

If several `emsg` boxes are present in a segment and one of them is the MPD Validity Expiration event, `emsg` carrying it shall always appear first.

5.3.2.4. MPD events

In addition to tracking events (ad starts, quartile tracking, etc.) the server may also need to signal additional metadata to the video application. For example, an ad unit may contain not only inline linear ad content (that is to be played before, during, or after the main presentation), it may also contain a companion display ad that is to be shown at the same time as the video ad. It is important that the server be able to signal both the presence of the companion ad and the additional tracking and click-through metadata associated with the companion.

With that said, there is no need to have a generic DASH client implement this functionality – it is enough to provide opaque information that the client would pass to an external module. Event `@schemeIdUri` provides us with such addressing functionality, while MPD events allow us to put opaque payloads into the MPD.

5.3.3. Workflows

5.3.3.1. General

In the workflows below we assume that our inputs are MPEG-2 transport streams with embedded SCTE 35 cue messages [54]. In our opinion this will be a frequently encountered deployment, however any other in-band or out-of-band method of getting cue messages and any other input format lend themselves into the same model.

5.3.3.2. Linear

A real-time MPEG-2 TS feed arrives at both packager and MPD generator. While real-time multicast feeds are a very frequently encountered case, the same workflow can apply to cases such as ad replacement in a pre-recorded content (e.g., in time-shifting or PVR scenarios).

MPD generator generates dynamic MPDs. Packager creates DASH segments out of the arriving feed and writes them into the origin server. Client periodically requests the MPDs so that it has enough time to transition seamlessly into the ad period.

Packager and MPD generator may be tightly coupled (e.g. co-located on the same physical machine), or loosely coupled as they both are synchronized only to the clock of the feed.

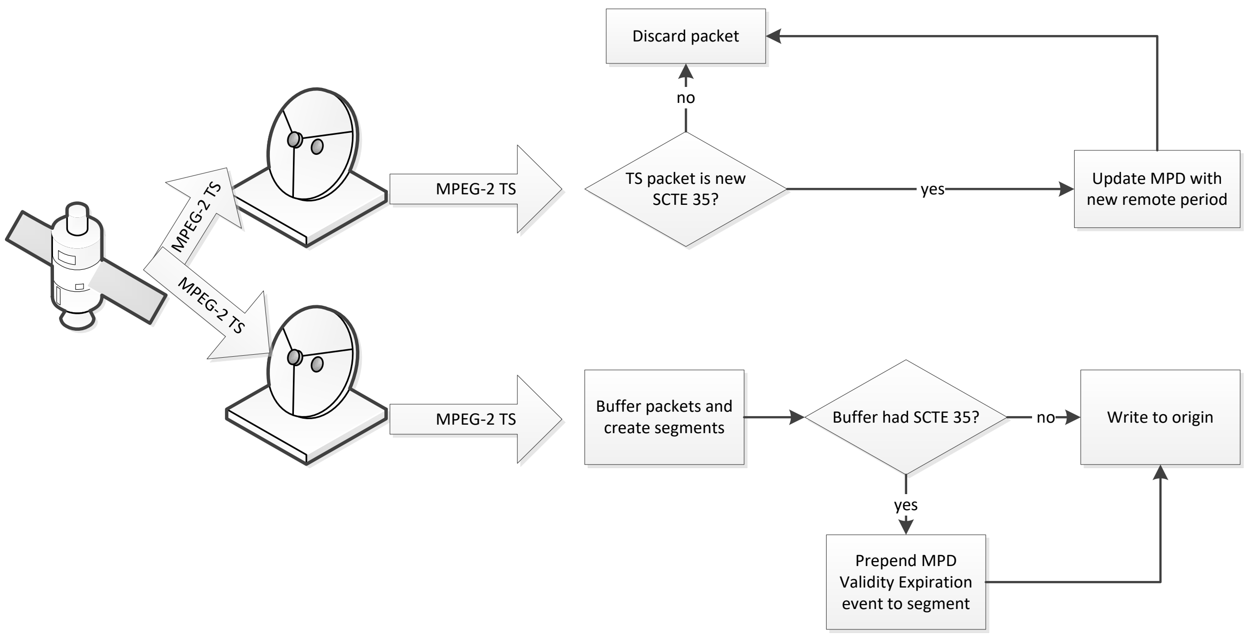

Figure 14: Live Workflow

5.3.3.2.1. Cue Interpretation by the MPD generator

When an SCTE 35 cue message indicating an upcoming splice point is encountered by the MPD generator, the latter creates a new MPD for the same program, adding a remote period to it.

The **Period**@start attribute of the inserted period has `splice_time()` translated into the presentation timeline. Parameters derived from the cue message are inserted into the **Period**@xlink:href attribute of the inserted period. Examples below show architectures that allow finer targeting.

5.3.3.2.1.1. Example 1: Immediate ad decision

MPD generator keeps an up-to-date template of an MPD. At each cue message arrival, the generator updates its template. At each MPD request, the generator customizes the request based on the information known to it about the requesting client. The generator contacts ad decision server and produces one or more non-remote ad periods. In this case XLink is not needed.

5.3.3.2.1.2. Example 2: Stateful cue translation

MPD generator keeps an up-to-date template of an MPD. At each cue message arrival, the generator updates its template. At each MPD request, the generator customizes the request based on the information known to it about the requesting client.

The operator targets separately male and female audiences. Hence, the generator derives this from the information it has regarding the requesting client (see 5.1.3.6), and inserts an XLink URL with the query parameter `?gender=male` for male viewers, and `?gender=female` for the female viewers.

Note that this example also showcases poor privacy practices – would such approach be implemented, both parameter name and value should be encrypted or TLS-based communication should be used

5.3.3.2.1.3. Example 3: Stateless cue translation

At cue message arrival, the MPD generator extracts the entire SCTE 35 `splice_info_section` (starting at the `table_id` and ending with the `CRC_32`) into a buffer. The buffer is then encoded into URL-safe base64url format according to RFC 4648 [60], and inserted into the XLink URL of a new remote Period element. `splice_time` is translated into **Period**`@start` attribute. The new MPD is pushed to the origin.

> Note: this example is a straightforward port of the technique defined for SCTE 67 [55], but uses base64url and not base64 encoding as the section is included in a URI.

5.3.3.2.2. Cue Interpretation by the packager

Cue interpretation by the packager is optional and is an optimization, rather than core functionality. On reception of an SCTE 35 cue message signaling an upcoming splice, an `` `emsg` `` with MPD Validity Expiration event is inserted into the first available segment. This event triggers an MPD update, and not an ad decision, hence the sum of the earliest presentation time of the `` `emsg` ``-bearing segment and the `` `emsg`.presentation_time_delta `` should be sufficiently earlier than the splice time. This provides the client with sufficient time to both fetch the MPD and resolve XLink.

`splice_time()` of the cue message is translated into the media timeline, and last segment before the splice point is identified. If needed, the packager can also finish the segment at the splice point and thus having a segment shorter than its target duration.

5.3.3.2.3. Multiple cue messages

There is a practice of sending several SCTE 35 cue messages for the same splice point (e.g., the first message announces a splice in 6 seconds, the second arrives 2 seconds later and warns about the same splice in 4 seconds, etc.). Both the packager and the MPD generator react on the same first message (the 6-sec warning in the example above), and do nothing about the following messages.

5.3.3.2.4. Cancelation

It is possible that the upcoming (and announced) insertion will be canceled (e.g., ad break needed to be postponed due to overtime). Cancelation is announced in a SCTE 35 cue message.

When cancelation is announced, the packager will insert the corresponding `` `emsg` `` event and the MPD generator will create a newer version of the MPD that does not contain the inserted period or sets its duration to zero. This implementation maintains a simpler less-coupled server side system at the price of an increase in traffic.

5.3.3.2.5. Early termination

It is also possible that a planned ad break will need to be cut short – e.g., an ad will be cut short and there will be a switch to breaking news. The DASH translation of this would be creating an `` `emsg` `` at the packager and updating the MPD appropriately. Treatment of early termination here would be same as treatment of a switch from main content to an ad break.

It is easier to manipulate durations when **Period**`@duration` is absent and only **Period**`@start` is used – this way attributes already known to the DASH client don't change.

5.3.3.2.6. Informational cue messages

SCTE 35 can be used for purposes unrelated to signaling of placement opportunities. Examples of such use are content identification and time-of-day signaling. Triggering MPD validity expiration and possibly XLink resolution in this case may be an overreaction.

5.3.3.2.7. Ad decision

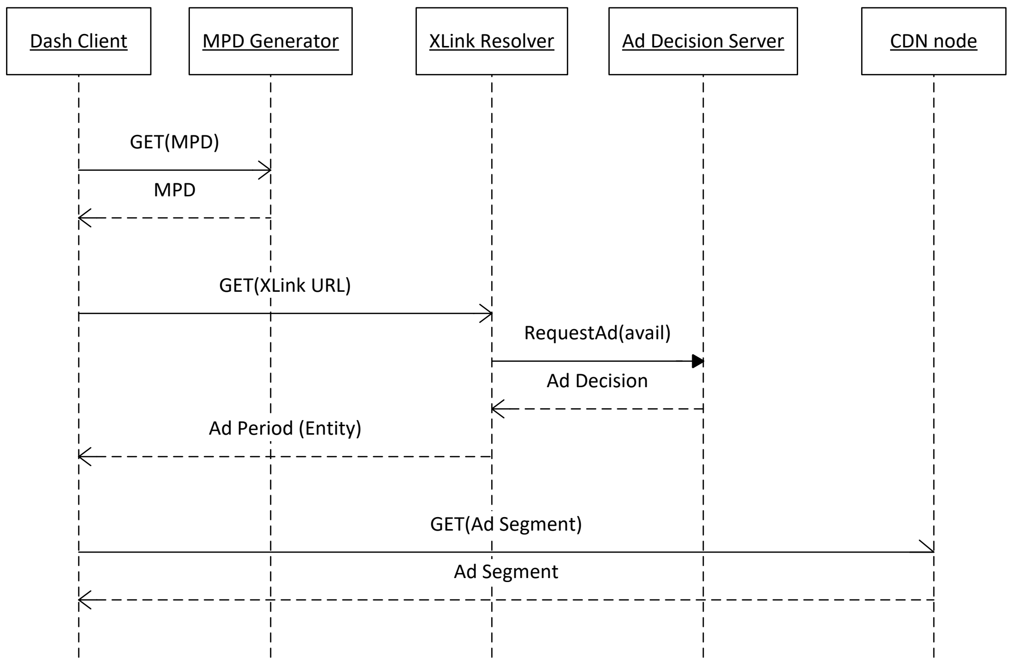

Figure 15: Ad Decision

A client will attempt to dereference a remote period element by issuing an HTTP GET for the URL that appears in **`Period`**`@xlink:href`. The HTTP server responding to this request (XLink resolver) will contact the ad decision service, possibly passing it parameters known from the request URL and from client information available to it from the connection context. In case described in 5.3.3.2.1.3, the XLink resolver has access to a complete SCTE 35 message that triggered the splice.

The ad decision service response identifies the content that needs to be presented, and given this information the XLink resolver can generate one or more Period elements that would be then returned to the requesting DASH client.

A possible optimization is that resolved periods are cached – e.g. in case of 5.3.3.2.1.1 "male" and "female" versions of the content are only generated once in *T* seconds, with HTTP caching used to expire the cached periods after *T* seconds.

5.3.3.3. On Demand

In a VoD scenario, cue locations are known ahead of time. They may be available multiplexed into the mezzanine file as SCTE 35 or SCTE 104, or may be provided via an out-of-band EDL.

In VoD workflows both cue locations and break durations are known, hence there is no need for a dynamic MPD. Thus cue interpretation (which is same as in 5.3.3.2) can occur only once and result in a static MPD that contains all remote elements with all Period elements having **Period**@start attribute present in the MPD.

In elastic workflows ad durations are unknown, thus despite our knowledge of cue locations within the main content it is impossible to build a complete presentation timeline. **Period**@duration needs to be used. Remote periods should be dereferenced only when needed for playout. In case of a "jump" – random access into an arbitrary point in the asset – it is a better practice not to dereference Period elements when it is possible to determine the period from which the playout starts using **Period**@duration and asset identifiers. The functionality described in 5.3.3.2 is sufficient to address on-demand cases, with the only difference that a client should be able to handle zero-duration periods that are a result of avails that are not taken.

5.3.3.4. Capture to VoD

Capture to VoD use case is a hybrid between pure linear and on demand scenarios: linear content is recorded as it is broadcast, and is then accessible on demand. A typical requirement is to have the content available with the original ad for some time, after which ads can be replaced

There are two possible ways of implementing the capture-to-VoD workflow.

The simplest is treating capture-to-VoD content as plain VoD, and having the replacement policy implemented on the XLink resolver side. This way the same Period element(s) will be always returned to the same requester within the window where ad replacement is disallowed; while after this window the behavior will be same as for any on-demand content. An alternative implementation is described in 5.3.3.5 below.

5.3.3.5. Slates and ad replacement

A content provider (e.g., OTT) provides content with ad breaks filled with its own ads. An ISP is allowed to replace some of these with their own ads. Conceptually there is content with slates in place of ads, but all slates can be shown and only some can be replaced.

An ad break with a slate can be implemented as a valid in-MPD Period element that also has XLink attributes. If a slate is replaceable, XLink resolution will result in new Period element(s), if not – the slate is played out.

5.3.3.6. Blackouts and Alternative content

In many cases broadcast content cannot be shown to a part of the audience due to contractual limitations (e.g., viewers located close to an MLB game will not be allowed to watch it, and will be shown some alternative content). While unrelated to ad insertion per se, this use case can be solved using the same "default content" approach, where the in-MPD content is the game and the alternative content will be returned by the XLink resolver if the latter determines (in some unspecified way) that the requester is in the blackout zone.

5.3.3.7. Tracking and reporting

A Period, either local or a remote entity, may contain an EventStream element with an event containing IAB VAST 3.0 Ad element [53]. DASH client does not need to parse the information and act accordingly – if there is a listener to events of this type, this listener can use the VAST 3.0 Ad element to implement reporting, tracking and companion ads. The processing done by this listener does not have any influence on the DASH client, and same content would be presented to both "vanilla" DASH client and the player in which a VAST module registers with a DASH client a listener to the VAST 3.0 events. VAST 3.0 response can be carried in an **Event** element where

EventStream@schemeIdUri value is http://dashif.org/identifiers/vast30.

An alternative implementation uses DASH Callback events to point to the same tracking URLs. While DASH specification permits both inband and MPD Callback events, inband callback events shall not be used.

5.3.4. Examples

5.3.4.1. MPD with mid-roll ad breaks and default content

In this example, a movie (“Top Gun”) is shown on a linear channel and has two mid-roll ad breaks. Both breaks have default content that will be played if the XLink resolver chooses not to return new Period element(s) or fails.

In case of the first ad break, SCTE 35 cue message is passed completely to the XLink resolver, together with the corresponding presentation time.

In case of the second ad break, proprietary parameters u and z describe the main content and the publishing site.

```
<?xml version="1.0"?>
<MPD xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="urn:mpeg:dash:schema:mpd:2011"
    xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 DASH-MPD.xsd"
    type="dynamic"
    minimumUpdatePeriod="PT2S"
    timeShiftBufferDepth="PT600S"
    minBufferTime="PT2S"
    profiles="urn:mpeg:dash:profile:isoff-live:2011"
    availabilityStartTime="2012-12-25T15:17:50">
    <BaseURL>http://cdn1.example.com/</BaseURL>
    <BaseURL>http://cdn2.example.com/</BaseURL>

    <!-- Movie -->
    <Period start="PT0.00S" duration="PT600.6S" id="movie period #1">
        <AssetIdentifier schemeIdUri="urn:org:dashif:asset-id:2013"
            value="md:cid:EIDR:10.5240%2f0EFB-02CD-126E-8092-1E49-W"/>
        <AdaptationSet mimeType="video/mp4" codecs="avc1.640828"
            frameRate="24000/1001" segmentAlignment="true" startWithSAP="1">
            <BaseURL>video_1/</BaseURL>
            <SegmentTemplate timescale="90000" initialization="$Band-
width%/init.mp4v"
                media="$Bandwidth$/$Number%05d$.mp4v"/>
            <Representation id="v0" width="320" height="240" bandwidth="250000"/>
            <Representation id="v1" width="640" height="480" bandwidth="500000"/>
            <Representation id="v2" width="960" height="720" bandwidth="1000000"/>
        </AdaptationSet>
    </Period>

    <!-- Mid-roll advertisement, passing base64url-coded SCTE 35 to XLink resolver -
->
    <Period duration="PT60.6S" id="ad break #1"
            xlink:href="https://adserv.com/avail.mpd?scte35-time=PT600.6S&
                 scte35-cue=DAIAAAAAAAAAAAQAAZ_I0VniQAQAgBDVUVJQAAAAH+cAAAAAA%3D%3D"
            xlink:actuate="onRequest" >

        <!-- Default content, replaced by elements from remote entity -->
        <AdaptationSet mimeType="video/mp4" codecs="avc1.640828"
```

```
                       frameRate="30000/1001"
            segmentAlignment="true" startWithSAP="1">
            <BaseURL availabilityTimeOffset="INF">default_ad/</BaseURL>
            <SegmentTemplate timescale="90000" initialization="$Band-
width%/init.mp4v"
                media="$Bandwidth%/$Time$.mp4v"/>
            <Representation id="v0" width="320" height="240" bandwidth="250000"/>
            <Representation id="v1" width="640" height="480" bandwidth="500000"/>
            <Representation id="v2" width="960" height="720" bandwidth="1000000"/>
        </AdaptationSet>
    </Period>

    <!—Movie, cont'd -->
    <Period duration="PT600.6S" id="movie period #2">
        <AssetIdentifier schemeIdUri="urn:org:dashif:asset-id:2013"
            value="md:cid:EIDR:10.5240%2f0EFB-02CD-126E-8092-1E49-W"/>
        <AdaptationSet mimeType="video/mp4" codecs="avc1.640828"
                       frameRate="24000/1001"
            segmentAlignment="true" startWithSAP="1">
            <BaseURL>video_2/</BaseURL>
            <SegmentTemplate timescale="90000" initialization="$Band-
width%/init.mp4v"
                media="$Bandwidth%/$Time$.mp4v"/>
            <Representation id="v0" width="320" height="240" bandwidth="250000"/>
            <Representation id="v1" width="640" height="480" bandwidth="500000"/>
            <Representation id="v2" width="960" height="720" bandwidth="1000000"/>
        </AdaptationSet>
    </Period>


    <!-- Mid-roll advertisement, using proprietary parameters  -->
    <Period duration="PT60.6S" id="ad break #2"
        xlink:href=”https://adserv.com/avail.mpd?u=0EFB-02CD-126E-8092-1E49-
W&z=spam”
        xlink:actuate="onRequest" >

        <!-- Default content, replaced by elements from remote entity -->
        <AdaptationSet mimeType="video/mp4" codecs="avc1.640828"
                       frameRate="30000/1001"
            segmentAlignment="true" startWithSAP="1">
            <BaseURL availabilityTimeOffset="INF">default_ad2/</BaseURL>
            <SegmentTemplate timescale="90000" initialization="$Band-
width%/init.mp4v"
                media="$Bandwidth%/$Time$.mp4v"/>
            <Representation id="v0" width="320" height="240" bandwidth="250000"/>
            <Representation id="v1" width="640" height="480" bandwidth="500000"/>
            <Representation id="v2" width="960" height="720" bandwidth="1000000"/>
        </AdaptationSet>
    </Period>
</MPD>
```

Figure 16: Example of MPD for "Top Gun" movie

5.3.5. Use of query parameters

Parameters can be passed into the XLink resolver as a part of the XLink URL. Clause 5.3.3.2.1.3 shows an example of this approach when an SCTE 35 cue message is embedded into the XLink URL.

This approach can be generalized and several parameters (i.e., name-value pairs) can be defined. SCTE 214-1 2016 [56] takes this approach and defines parameters expressing splice time (i.e.,

Period@start of the earliest ad period), SCTE 35 cue message, and syscode (a geolocation identifier used in US cable industry). The first two parameters are also shown in example in clause 5.3.4.1 of this document.

> Note 1: Effectively this creates a RESTful API for XLink dereferencing. While discussion above implies that these parameters are embedded by the MPD generator into the XLink URL, the parameter values may as well be calculated by the client or the embedded values may be modified by the client.
> Note 2: The same RESTful API approach can be used with MPD URLs as well.
> Note 3: More parameters may be defined in the future version of these guidelines.

5.4. App-based Architecture

5.4.1. Introduction

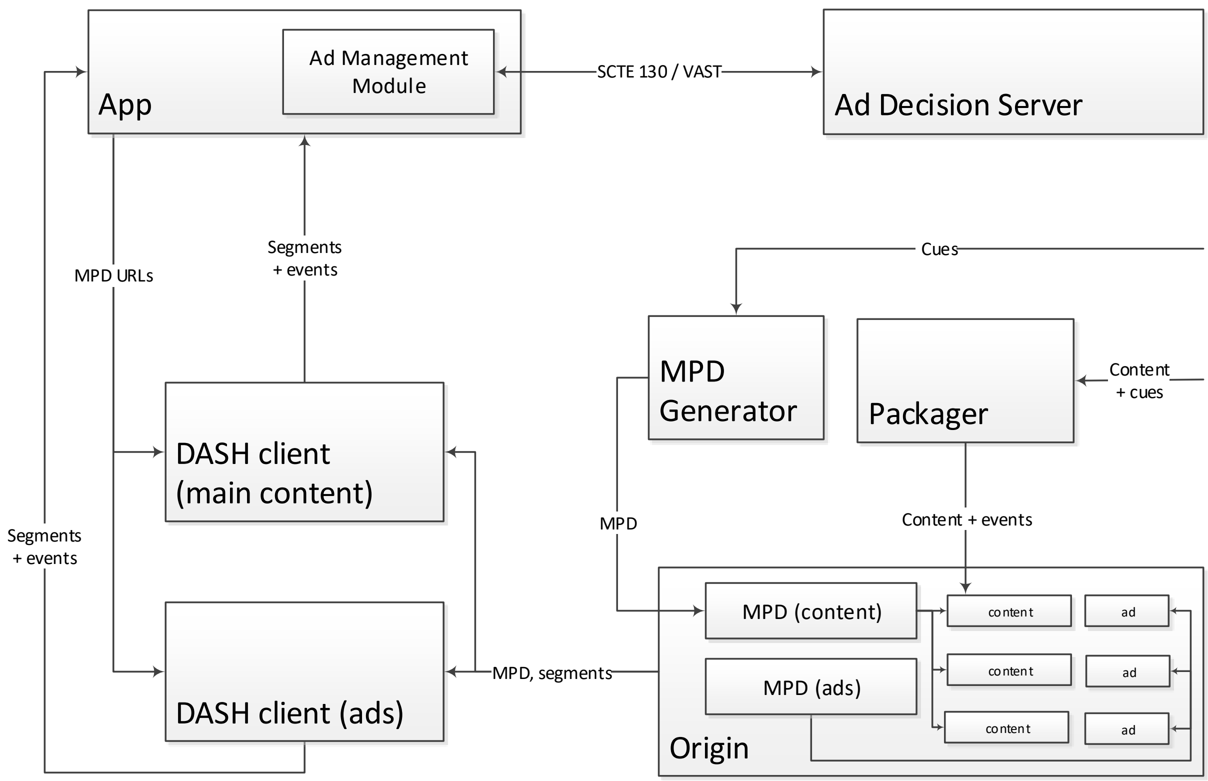

Figure 17: App-based architecture

Inputs in this use case are same as the ones described in sec. 5.3. At the packaging stage, cues are translated into a format readable by the app or/and DASH client and are embedded into media segments or/and into the manifest

Ad management module is located at the client side. The DASH client receives manifest and segments, with cues embedded in either one of them or in both.

Cue data is passed to the ad management module, which contacts the ad decision service and receives information on content to be played. This results in an MPD for an inserted content and a splice time at which presentation of main content is paused and presentation of the inserted content starts.

Note that this architecture does not assume multiple decoders – with careful conditioning it is possible to do traditional splicing where inserted content is passed to the same decoder. In this case it is necessary to keep a player state and be able to initialize a player into this state.

5.4.2. Mapping to DASH

This section details mapping of elements of the reference architecture into DASH concepts per the 2nd edition of the specification (i.e., ISO/IEC 23009-1:2014).

5.4.2.1. MPD

Each ad decision results in a separate MPD. A single MPD contains either main content or inserted content; existence of multiple periods or/and remote periods is possible but not essential.

5.4.2.2. SCTE 35 events

5.4.2.2.1. General

Cue messages are mapped into DASH events, using inband `emsg` boxes and/or in-MPD events. Note that SCTE 35 cue message may not be sufficient by itself.

The examples below show use of SCTE 35 in user-defined events, and presentation time indicates the timing in within the Period.

Figure 18 below shows the content of an `emsg` box at the beginning of a segment with earliest presentation time *T*. There is a 6-sec warning of an upcoming splice – delta to splice time is indicated as 6 seconds – and duration is given as 1 minute. This means that an ad will start playing at time $T + 6$ till $T + 66$. This example follows a practice defined in SCTE DVS 1208.

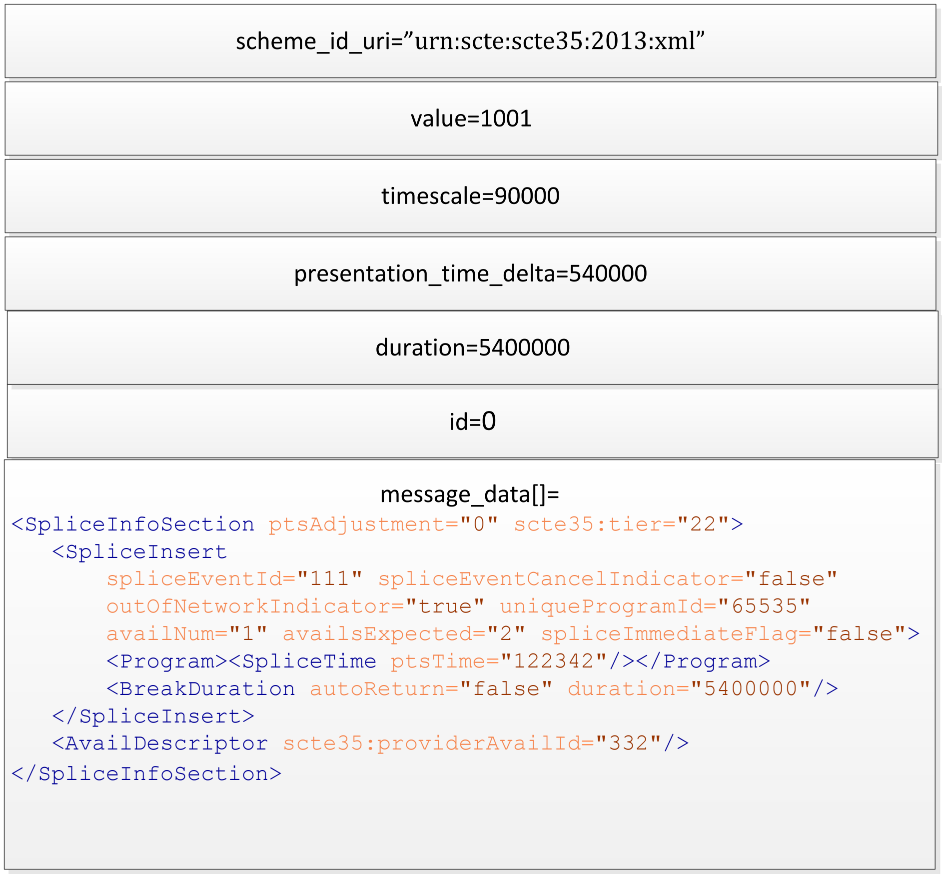

Figure 18 Inband carriage of SCTE 35 cue message

Figure 19 below shows the same example with an in-MPD SCTE35 cue message. The difference is in the in-MPD event the splice time is relative to the Period start, rather than to the start of the event-carrying segment. This figure shows a one-minute ad break 10 minutes into the period.

```
<EventStream schemeIdUri="urn:scte:scte35:2014:xml+bin">
    <Event timescale="90000" presentationTime="54054000" duration="5400000" id="1">
        <scte35:Signal>
             <scte35:Binary>
                 /DAIAAAAAAAAAAAQAAZ/I0VniQAQAgBDVUVJQAAAAH+cAAAAAA==
             </scte35:Binary>
         </scte35:Signal>
    </Event>
</EventStream>
```

Figure 19: In-MPD carriage of SCTE 35 cue message

Note: for brevity purposes SCTE 35 2014 allows use of base64-encoded section in **`Signal.Binary`** element as an alternative to carriage of a completely parsed cue message.

Normative definitions of carriage of SCTE 35 cue messages are in ANSI/SCTE 214-1 sec 6.8.4 (MPD) and ANSI/SCTE 214-3 sec 8.3.3.

5.4.2.3. Asset Identifiers

See sec. 5.3.2.2 for details.

5.4.3. Workflows

5.4.3.1. Linear

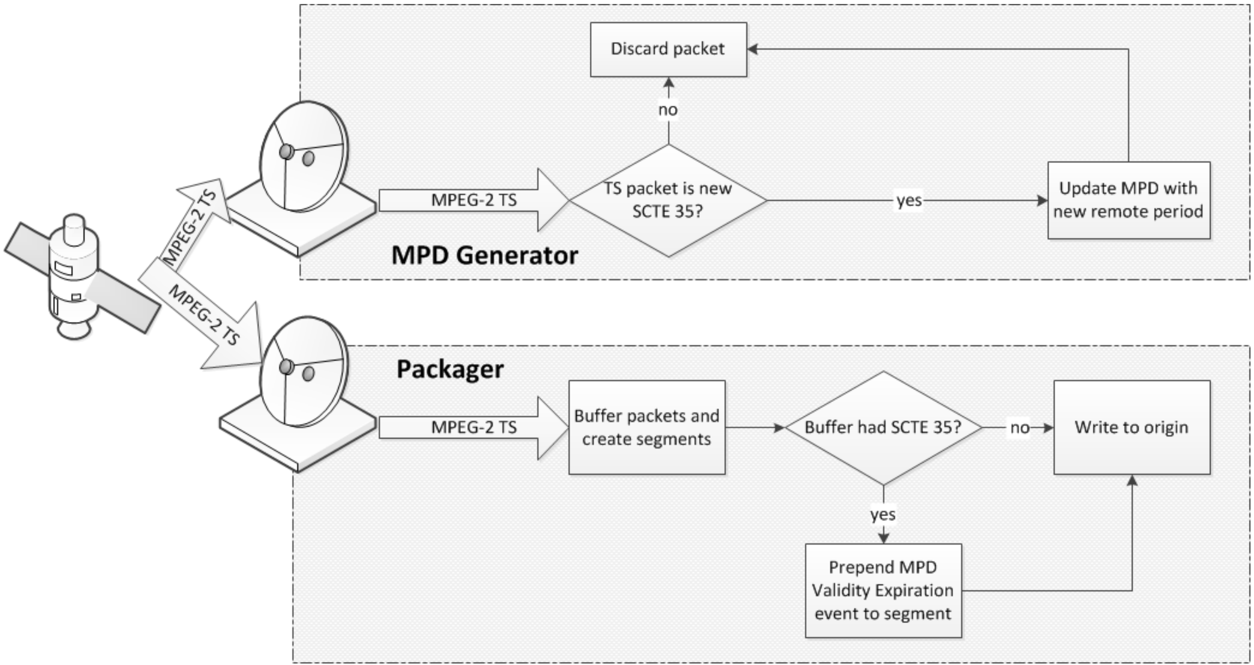

Figure 20: Linear workflow for app-driven architecture

A real-time MPEG-2 TS feed arrives at a packager. While real-time multicast feeds are a very frequently encountered case, the same workflow can apply to cases such as ad replacement in a pre-recorded content (e.g., in time-shifting or PVR scenarios).

Packager creates DASH segments out of the arriving feed and writes them into the origin server. The packager translates SCTE 35 cue messages into inband DASH events, which are inserted into media segments.

MPD generator is unaware of ad insertion functionality and the packager does the translation of SCTE 35 cue messages into inband user-defined DASH events. On reception of an SCTE 35 cue message signaling an upcoming splice, a `` `emsg` `` with a translation of the cue message in its `` `emsg`.message_data[] `` field is inserted into the most recent Segment. This event triggers client interaction with an ad decision server, hence the sum of the earliest presentation time of the `` `emsg` ``-bearing segment and the `` `emsg`.presentation_time_delta `` should be a translation of `splice_time()` into the media timeline.

An alternative implementation which is more compatible with server-based architecture in section 5.3, an MPD generator can generate separate MPDs for both server-based and app-based architectures creating remote periods for server-based and in-MPD SCTE 35 events for app-based architectures, while a packager can insert inband MPD validity expiration events.

A DASH client will pass the event to the app controlling it (e.g., via a callback registered by the app). The app will interpret the event and communicate with the ad decision server using some interface (e.g., VAST). This interface is out of the scope of this document.

The communication with ad decision service will result in an MPD URL. An app will pause the presentation of the main content and start presentation of the inserted content. After presenting the inserted content the client will resume presentation of the main content. This assumes either proper conditioning of the main and inserted content or existence of separate client and decoder for inserted content. The way pause/resume is implemented is internal to the API of the DASH client. Interoperability may be achieved by using the DASH MPD fragment interface, see ISO/IEC 23009-1 [4], Annex C.4.

5.4.3.2. On Demand

As in the server-based case, functionality defined for the live case is sufficient. Moreover, the fact that that app-based implementation relies heavily on app's ability to pause and resume the DASH client, support for elastic workflows is provided out of the box.

In the on demand case, as cue locations are well-known, it is advantageous to provide a static MPD with SCTE 35 events than run a dynamic service that relies on inband events.

5.5. Extensions for ad insertion

5.5.1. Asset Identifiers

`AssetIdentifier` descriptor shall be used for distinguishing parts of the same asset within a multi-period MPD, hence it shall be used for main content and may be used for inserted content.

In order to enable better tracking and reporting, unique IDs should be used for different assets.

Use of EIDR and Ad-ID identification schemes is recommended. The value of `@schemeIdUri` set to "`urn:eidr`" signals use of EIDR. The value of `@value` attribute shall be a valid canonical EIDR entry as defined in [67].

Use of Ad-ID for asset identification is signaled by setting the value of @schemeIdUri to "`urn:smpte:ul:060E2B34.01040101.01200900.00000000`" ("`designator`" URN defined in SMPTE 2092-1 [68]). The value of `@value` attribute shall be a canonical full Ad-ID identifier as defined in SMPTE 2092-1 [68].

Other schemes may be used, including user private schemes, by using appropriately unique values of `@schemeIdUri`.

In the absence of other asset identifier schemes, a DASH-IF defined scheme may be used with the value of `@schemeIdUri` set to "`urn:org:dashif:asset-id:2014`". If used, the value of `@value` attribute descriptor shall be a MovieLabs ContentID URN ([58], 2.2.1) for the content. It shall be the same for all parts of an asset. Preferred schemes are EIDR (main content) and Ad-ID (advertising).

If a Period has one-off semantics (i.e., an asset is completely contained in a single period, and its continuation is not expected in the future), the author shall not use asset identifier on these assets.

Periods that do not contain non-remote **`AdaptationSet`** elements, as well as zero-length periods shall not contain the **`AssetIdentifier`** descriptor.

5.5.2. Remote Periods

An MPD may contain remote periods, some of which may have default content. Some of which are resolved into multiple Period elements.

After dereferencing MPD may contain zero-length periods or/and remote Periods.

In case of **Period**@xlink:actuate="onRequest", MPD update and XLink resolution should be done sufficiently early to ensure that there are no artefacts due to insufficient time given to download the inserted content.

Period@xlink:actuate="onRequest" shall not be used if **MPD**@type ="dynamic"

5.5.3. User-defined events

5.5.3.1. Cue message

Cue messages used in app-driven architecture shall be SCTE 35 events [54]. SCTE 35 event carriage is defined in ANSI/SCTE 214-1 (MPD) and ANSI/SCTE 214-3 (inband). For MPD events, the XML schema is defined in SCTE 35 2014 [54] and allows either XML representation or concise base64-coded representation.

NOTE: PTS offset appearing in SCTE 35 shall be ignored, and only DASH event timing mechanism may be used to determine splice points.

5.5.3.2. Reporting

MPD events with embedded IAB VAST 3.0 [53] response may be used for reporting purposes.

If only time-based reporting is required (e.g., reporting at start, completion, and quartiles), use of DASH callback event may be a simpler native way of implementing tracking. Callback events are defined in ISO/IEC 23009-1:2014 AMD3 [4].

5.5.3.3. Ad Insertion Event Streams

Recommended Event Stream schemes along with their scheme identifier for app-driven ad insertion are:

1. "urn:scte:scte35:2013:bin" for inband SCTE 35 events containing a complete SCTE 35 section in binary form, as defined in ANSI/SCTE 214-3.
2. "urn:scte:scte35:2014:xml+bin" for SCTE 35 MPD events containing only base64 cue message representation, as defined in ANSI/SCTE 214-1.
   NOTE: the content of Event element is an XML representation of the complete SCTE 35 cue message, that contains **Signal.Binary** element rather than the **Signal.SpliceInfoSection** element, both defined in SCTE 35 2014.
3. "http://dashif.org/identifiers/vast30" for MPD events containing VAST3.0 responses [53].
4. urn:mpeg:dash:event:callback:2015 for DASH callback events.

5.6. Interoperability Aspects

5.6.1. Server-based Ad insertion

For server-based ad insertion, the following aspects needs to be taken into account:

- Service offerings claiming conformance to server-based ad insertion shall follow the requirements and guidelines for service offerings in sections 5.3.2, 5.5.1, and 5.5.2..
- Clients claiming conformance to server-based ad insertion shall follow shall follow the requirements and guidelines for clients in section 5.3.2, 5.5.1, and 5.5.2. .

5.6.2. App-based Ad Insertion

For app-based ad insertion, the logic for ad insertion is outside the scope of the DASH client. The tools defined in section 5.4 and 5.5 may be used to create an interoperable system that includes DASH-based delivery and ad insertion logic.

6. Media Coding Technologies

6.1. Introduction

In addition to DASH-specific constraints, DASH-IF IOPs also adds restrictions on media codecs and other technologies. This section provides an overview on technologies for different media components and how they fit into the DASH-related aspects of DASH-IF IOPs.

6.2. Video

6.2.1. General

The codec considered for basic video support up to 1280 x 720p at 30 fps is H.264 (AVC) Progressive High Profile Level 3.1 decoder [8]. This choice is based on the tradeoff between content availability, support in existing devices and compression efficiency.

Further, it is recognized that certain clients may only be capable to operate with H.264/AVC "Progressive" Main Profile Level 3.0 and therefore content authors may provide and signal a specific subset of DASH-IF IOP.

Notes

- H.264 (AVC) Progressive High Profile Level 3.1 decoder [8] can also decode any content that conforms to
  - H.264 (AVC) Constrained Baseline Profile up to Level 3.1
  - H.264 (AVC) "Progressive" Main Profile up to Level 3.1.
- H.264 (AVC) H.264/AVC "Progressive" Main Profile Level 3.0 decoder [8] can also decode any content that conforms to H.264 (AVC) Constrained Baseline Profile up to Level 3.0.

Further, the choice for HD extensions up to 1920 x 1080p and 30 fps is H.264 (AVC) Progressive High Profile Level 4.0 decoder [8].

The High Efficiency Video Coding (HEVC) resulted from a joint video coding standardization project of the ITU-T Video Coding Experts Group (ITU-T Q.6/SG 16) and ISO/IEC Moving Picture Experts Group (ISO/IEC JTC 1/SC 29/WG 11). The final specification is available here [19]. Additional background information may be found at http://hevc.info.

The DASH-IF is interested in providing Interoperability Points and Extensions for established codec configurations. It is not the intent of the DASH-IF to define typically deployed HEVC profiles/levels or the associated source formats. However, at the same time it is considered to provide implementation guidelines supported by test material for DASH-based delivery as soon as the industry has converged to profile/level combinations in order to support a dedicated format. For this version of this document the following is considered:

- For HEVC-based video, it is expected that the minimum supported format is 720p. The codec considered to support up to 1280 x 720p at 30 fps is HEVC Main Profile Main Tier Level 3.1 [19].

- The choice for 8-bit HD extensions based on HEVC to support up to 2048 x 1080 and 60 fps is HEVC Main Profile Main Tier Level 4.1 [19].
- The choice for 10-bit HD extensions based on HEVC to support up to 2048 x 1080 and 60 fps and 10 bit frame depth is HEVC Main10 Profile Main Tier Level 4.1 [19].
- For UHD extensions refer to section 10.

Other profile/level combinations will be considered in updated versions of this document.

6.2.2. DASH-specific aspects for H.264/AVC video

For the integration of the above-referred codecs in the context of DASH, the following applies for H.264 (AVC):

- The encapsulation of H.264/MPEG-4 AVC video data is based on the ISO BMFF as defined in ISO/IEC 14496-15 [9].
- Clients shall to support H.264/AVC sample entries when SPS/PPS is provided in the Initialization Segment only according to ISO/IEC 14496-15, [9], i.e. sample entry '`avc1`'.
- Clients shall support Inband Storage for SPS/PPS based ISO/IEC 14496-15, [9], i.e. sample entry '`avc3`'.
- Service offerings using H.264/AVC may use sample entry '`avc1`' or '`avc3`'.
- SAP types 1 and 2 correspond to IDR-frames in [8].
- The signaling of the different video codec profile and levels for the codecs parameters according to RFC6381 [10] is documented in Table 17. Note that any of the codecs present in Table 17 conforms to the profile level combination that is supported in DASH-AVC/264. Other codecs strings may be used and conform as well.
- Additional constraints within one Adaptation Set are provided in section 6.2.5.

  Note: For a detailed description on how to derive the signaling for the codec profile for H.264/AVC, please refer to DVB DASH, section 5.1.3.

Table 17 H.264 (AVC) Codecs parameter according to RFC6381 [10]

Profile	Level	Codec Parameter
H.264 (AVC) "Progressive" Main Profile	3.0	`avc[1,3].4DY01E`
H.264 (AVC) Progressive High Profile	3.1	`avc[1,3].64Y01F`
	4.0	`avc[1,3].64Y028`

6.2.3. DASH-specific aspects for H.265/HEVC video

For the integration in the context of DASH, the following applies for HEVC

- The encapsulation of HEVC video data in ISO BMFF is defined in ISO/IEC 14496-15 [9]. Clients shall support both sample entries ' using 'hvc1' and 'hev1', i.e.. inband Storage for VPS/SPS/PPS.
- Additional constraints within one Adaptation Set are provided in section 6.2.5.

- For the signaling of HEVC IRAP Pictures in the ISOBMFF and in DASH, in particular the use of the sync sample table and of the SAP sample group, please refer to Table 18.

Table 18 Signaling of HEVC IRAP Pictures in the ISOBMFF and in DASH

NAL Unit Type	ISOBMFF sync status	DASH SAP type
IDR_N_LP	true	1
IDR_W_RADL	true	2 (if the IRAP has associated RADL pictures) 1 (if the IRAP has no associated RADL pictures)
BLA_N_LP	true	1
BLA_W_RADL	true	2 (if the IRAP has associated RADL pictures) 1 (if the IRAP has no associated RADL pictures)
BLA_W_LP	false	3 (if the IRAP has associated RASL pictures)
	true	2 (if the IRAP has no associated RASL pictures but has associated RADL pictures)
	true	1 (if the IRAP has no associated leading pictures)
CRA	false	3 (if the IRAP has associated RASL pictures)
	true	2 (if the IRAP has no associated RASL pictures but has associated RADL pictures)
	true	1 (if the IRAP has no associated leading pictures)
In the above table, when there are multiple possible values for a given NAL Unit Type, if the entity creating the signaling is not able to determine correctly which signaling to use, it shall use the values in the first row of this table associated to the NAL Unit Type.		

- The signaling of the different video codec profile and levels for the codecs parameters is according to ISO/IEC 14496-15 [9] Annex E. Note that any of the codecs present in Table 1 conforms to the profile level combination that is supported in DASH-HEVC.

NOTE: For a detailed description on how to derive the signaling for the codec profile for H.264/AVC, please refer to DVB DASH, section 5.2.2.

Table 19 Codecs parameter according to ISO/IEC 14496-15 [9]

Profile	Level	Tier	Codec Parameter
HEVC Main	3.1	Main	hev1.1.2.L93.B0 hvc1.1.2.L93.B0
	4.1	Main	hev1.1.2.L123.B0 hvc1.12.L123.B0
HEVC Main-10	4.1	Main	hev1.2.4.L123.B0 hvc1.2.4.L123.B0

6.2.4. Video Metadata

The provisioning of video metadata in the MPD is discussed in section 3.2.4.

6.2.5. Adaptation Sets Constraints

6.2.5.1. General

Video Adaptation Sets shall contain Representations that are alternative encodings of the same source content. Video Adaptation Sets may contain Representations encoded at lower resolutions that are exactly divisible subsamples of the source image size. As a result, the cropped vertical and horizontal sample counts of all Representations can be scaled to a common display size without position shift or aspect ratio distortion that would be visible during adaptive switching. Subsample ratios must result in integer values for the resulting encoded sample counts (without rounding or truncation). The encoded sample count shall scale to the source video's exact active image aspect ratio when combined with the encoded sample aspect ratio value `aspect_ratio_idc` stored in the video Sequence Parameter Set NAL. Only the active video area shall be encoded so that devices can frame the height and width of the encoded video to the size and shape of their currently selected display area without extraneous padding in the decoded video, such as "letterbox bars" or "pillarbox bars".

All decoding parameter sets referenced by NALs in a Representation using 'avc1' or 'hvc1' sample description shall be indexed to that track's sample description table and decoder configuration record in the '`avcC`' or '`hvcC`' box contained in its Initialization Segment. All decoding parameter sets referenced by NALs in a Representation using '`avc3`' or '`hev1`' sample description shall be indexed to a Sequence Parameter NAL (SPS) and Picture Parameter NAL (PPS) stored prior to the first video sample in that Media Segment. For '`avc3`' and '`hev1`' sample description Representations, the SPS and PPS NALs stored in '`avcC`' or '`hvcC`' in the Initialization Segment shall only be used for decoder and display initialization, and shall equal the highest Tier, Profile, and Level of any SPS in the Representation. SPS and PPS stored in each Segment shall be used for decoding and display scaling.

For all Representations within an Adaptation Set with the following parameters shall apply.

- All the Initialization Segments for Representations within an Adaptation Set shall have the same sample description `codingname`. For example the inclusion of '`avc1`' and '`avc3`' based Representations within an Adaptation Set or the inclusion '`avc1`' and '`hev1`' based Representations within an Adaptation Set is not permitted.
- All Representations shall have equal timescale values in all `@timescale` attributes and '`tkhd`' `timescale` fields in Initialization Segments.
- If '`avc1`' or '`hvc1`' sample description is signaled in the **`AdaptationSet`**`@codecs` attribute, an edit list may be used to synchronize all Representations to the presentation timeline, and the edit offset value shall be equal for all Representations.
- Representations in one Adaptation Set shall not differ in any of the following parameters: Color Primaries, Transfer Characteristics and Matrix Coefficients. If Adaptation Sets differ in any of the above parameters, these parameters should be signaled on Adaptation Set level. If signaled, a Supplemental or Essential Property descriptor shall be used, with the `@schemeIdUri` set to `urn:mpeg:mpegB:cicp:<Parameter>` as defined in ISO/IEC 23001-8 [49] and <Parameter> one of the following: `ColourPrimaries`,

TransferCharacteristics, or MatrixCoefficients. The @value attribute shall be set as defined in ISO/IEC 23001-8 [49].

6.2.5.2. Bitstream Switching

For AVC and HEVC video data, if the @bitstreamswitching flag is set to true, then the following additional constraints shall apply:

- All Representations shall be encoded using 'avc3' sample description for AVC or 'hev1' for HEVC, and all IDR pictures shall be preceded by any SPS and PPS NAL decoding parameter referenced by a video NAL in that codec video sequence.

  Note: NAL parameter indexes in a Media Segment are scoped to that Segment. NALs and indexes in the Initialization Segment may be different, and are only used for decoder initialization, not Segment decoding.
- All Representations within a video Adaptation Set shall include an Initialization Segment containing an 'avcC' or 'hvcC' Box containing a Decoder Configuration Record containing SPS and PPS NALs that equal the highest Tier, Profile, Level, vertical and horizontal sample count of any Media Segment in the Representation. HEVC Decoder Configuration Records shall also include a VPS NAL.
- The **AdaptationSet**@codecs attribute shall be present and equal the maximum profile and level of any Representation contained in the Adaptation Set.
- The **Representation**@codecs attribute may be present and in that case shall equal the maximum profile and level of any Segment in the Representation.
- Edit lists shall not be used to synchronize video to audio and presentation timelines.
- Video Media Segments shall set the first presented sample's composition time equal to the first decoded sample's decode time, which equals the baseMediaDecodeTime in the Track Fragment Decode Time Box ('tfdt').

  Note: This requires the use of negative composition offsets in a v1 Track Run Box ('trun') for video samples, otherwise video sample reordering will result in a delay of video relative to audio.
- The @presentationTimeOffset attribute shall be sufficient to align audio video, subtitle, and presentation timelines at presentation a Period's presentation start time. Any edit lists present in Initialization Segments shall be ignored. It is strongly recommended that the Presentation Time Offset at the start of each Period coincide with the first frame of a Segment to improve decoding continuity at the start of Periods.

NOTE: An Adaptation Set with the attribute **AdaptationSet**@bitstreamSwitching="true" fulfills the requirements of the DVB DASH specification [42].

See section 7.7 for additional Adaptation Set constraints related to content protection.

6.2.6. Tiles of thumbnail images

For providing easily accessible thumbnails with timing, Adaptation Sets with the new @contentType="image" may be used in the MPD. A typical use case is for enhancing a scrub bar with visual cues. The actual asset referred to is a rectangular tile of temporally equidistant thumbnails combined into one jpeg or png image. A tile, therefore is very similar to a video segment from MPD timing point of view, but is typically much longer. As for video, different spatial resolutions can be collected into one Adapation Set. To limit the implementation effort, only `SegmentTemplate` with `$Number$` is used to described the thumbnail tiles and their timing.

It is typically expected that the DASH client is able to process such Adaptation Sets by downloading the images and using browser-based processing to assign the thumbnails to the Media Presentation timeline.

A lot of parameters are the same as for video, but the ones which are new for thumbnail tiles, the rectangular grid dimensions are given as the value of the **`EssentialProperty`** with `@schemeIdUri` set to "`http://dashif.org/guidelines/thumbnail_tile`".

- If the **`EssentialProperty`** descriptor with `@schemeIdUri` set to [`http://dashif.org/guidelines/thumbnail_tile`](http://dashif.org/guidelines/thumbnail_tile) is present, the following attributes and elements of the Adaptation Set shall be used to describe the tiling as follows:The value of the descriptor provides the horizontal and vertical number of the tiles as unsigned integer, separated by an `'x'`. The two values are referred to as `htiles` and `vtiles` in the following.

- `@duration` expresses the duration of one tile in the media presentation timeline in the timescale provided by the value of the `@timescale` attribute, if present, otherwise in seconds. The value is referred to as `tduration` in the following.

- `@bandwidth` expresses the maximum tile size in bits divided by the duration of one tile as provided by the `tduration` value.

- `@width` and `@height` expresses the spatial resolution of the tile. Note that the maximum dimension of a JPEG image is 64k in width and height.

- Each tile has assigned a number starting with 1. The tile number is referred as `tnumber`.

- The `@startNumber` may be present to indicate the number of the first tile in the Period in the Period. If not present the first number is defaulted to 1. The value of the start number is referred to as `startnumber`.

- The `@presentationTimeOffset` may be present to indicate the presentation time of the thumbnail sequence at the start of the period. If not present, the first number is defaulted to 0. The timescale is provided by the value of the `@timescale` attribute, if present, otherwise in seconds. The value of the presentation time offset is referred to as `pto`.

Based on this information, the following information can be derived:

- information on how many thumbnails are included in one tile by multiplying the `htiles` with `vtiles`. This number is referred to as `ttiles`.

- The first tile that can be requested in the Period is referred to as `startnumber` which is used in the segment template.

- The presentation time in the period for each tile is defined as (`tnumber` – `1`)*`tduration - pto`.

- The duration of each thumbnail is defined as `tduration/ttiles`. This value is referred to as `thduration`.

- Thumbnail ordering is from left to right, row by row, starting from the top row. The last tile in a Period may have thumbnails outside the time interval. The content provider may use any padding pixel to such a tile, e.g. add black thumbnails. The number of a thumbnail in a tile is referred to as `thnumber`.

- The presentation time of the thumbnail within the tile is defined as (`thnumber` – 1)*`thduration`, i.e. the presentation within a period is defined as (`tnumber`-1)*`tduration + (thnumber` – 1)*`thduration - pto`.

- The vertical size of each thumbnail is defined as `@height/vtiles` and the horizontal size of each thumbnail is defined as `@width/htiles`.

- The total number of thumbnails in a Period, referred to tthumbnails is the ceiling of the period duration divided by the `thduration`.

An example Adaptation Set for tile-based thumbnails is provided below:

```
<AdaptationSet id="3" mimeType="image/jpeg" contentType="image">
  <SegmentTemplate     media="$RepresentationID$/tile$Number$.jpg”     duration="125"
startNumber="1"/>
  <Representation bandwidth="10000" id="thumbnails" width="6400" height="180">
    <EssentialProperty      schemeIdUri="http://dashif.org/guidelines/thumbnail_tile"
value="25x1"/>
  </Representation>
</AdaptationSet>
```

Here

- `htiles` is 25 and `vtiles` is set to 1. `ttiles` is derived as 25.
- `tduration` is 125 seconds and `thduration` is derived as 5 seconds.
- Assuming a Period duration of 899 seconds, the total number of thumbnails is 180.
- The vertical size of each thumbnail is 256 and the horizontal size is 180.
- The maximum bandwidth for each tile is 10 kbit/s.

6.3. Audio

6.3.1. General

Content offered according to DASH-IF IOP is expected to contain an audio component in most cases. Therefore, clients consuming DASH-IF IOP-based content are expected to support stereo audio. Multichannel audio support and support for additional codecs is defined in extensions in section 9 of this document.

The codec for basic stereo audio support is MPEG-4 High Efficiency AAC v2 Profile, level 2 [11].

Notes

- HE-AACv2 is also standardized as Enhanced aacPlus in 3GPP TS 26.401 [13].
- HE-AACv2 Profile decoder [8] can also decode any content that conforms to
  - MPEG-4 AAC Profile [11]
  - MPEG-4 HE-AAC Profile [11]

Therefore, Broadcasters and service providers encoding DASH-AVC/264 content are free to use any AAC version. It is expected that clients supporting the DASH-IF IOP interoperability point will be able to play AAC-LC, HE-AAC and HE-AACv2 encoded content.

For all HE-AAC and HE-AACv2 bitstreams, explicit backwards compatible signaling should be used to indicate the use of the SBR and PS coding tools.

> Note: To conform to the DVB DASH profile [42], explicit backwards compatible signaling shall be used to indicate the use of the SBR and PS coding tools.

For advanced audio technologies, please refer to section 9.

6.3.2. DASH-specific aspects for HE-AACv2 audio

In the context of DASH, the following applies for the High Efficiency AAC v2 Profile

- The content should be prepared according to the MPEG-DASH Implementation Guidelines [6] to make sure each (Sub)Segment starts with a SAP of type 1.
- The signaling of MPEG-4 High Efficiency AAC v2 for the codecs parameters is according to IETF RFC6381 [10] and is documented in Table 20. Table 20 also provides information on the ISO BMFF encapsulation.
- For content with SBR, i.e. `@codecs=mp4a.40.5` or `@codecs=mp4a.40.29`, `@audioSamplingRate` signals the resulting sampling rate after SBR is applied, e.g. 48 kHz even if the AAC-LC core operates at 24 kHz. For content with PS, i.e. `@codecs=mp4a.40.29`, **`AudioChannelConfiguration`** signals the resulting channel configuration after PS is applied, e.g. stereo even if the AAC-LC core operates at mono.

Table 20 HE-AACv2 Codecs parameter according to RFC6381 [10]

Codec	Codec Parameter	ISO BMFF Encapsulation	SAP type
MPEG-4 AAC Profile [11]	`mp4a.40.2`	ISO/IEC 14496-14 [12]	1

MPEG-4 HE-AAC Profile [11]	`mp4a.40.5`	ISO/IEC 14496-14 [12]	1
MPEG-4 HE-AAC v2 Profile [11]	`mp4a.40.29`	ISO/IEC 14496-14 [12]	1

Note: Since both, HE-AAC and HE-AACv2 are based on AAC-LC, for the above-mentioned “Codec Parameter” the following is implied:

- `mp4a.40.5` = `mp4a.40.2` + `mp4a.40.5`
- `mp4a.40.29` = `mp4a.40.2` + `mp4a.40.5` + `mp4a.40.29`

6.3.3. Audio Metadata

6.3.3.1. General

Metadata for audio services is defined in ISO/IEC 23009-1.

6.3.3.2. ISO/IEC 23009-1 audio data

With respect to the audio metadata, the following elements and attributes from ISO/IEC 23009-1 are relevant:

- the `@audioSamplingRate` attribute for signaling the sampling rate of the audio media component type in section 5.3.7 of ISO/IEC 23009-1
- the **`AudioChannelConfiguration`** element for signaling audio channel configuration of the audio media component type.in section 5.3.7 of ISO/IEC 23009-1. For this element the scheme and values defined in ISO/IEC 23001-8 for the **`ChannelConfiguration`** should be used.

6.4. Auxiliary Components

6.4.1. Introduction

Beyond regular audio and video support, TV programs typically also require support for auxiliary components such as subtitles and closed captioning, often due to regulatory requirements. DASH-IF IOP provides tools to addresses these requirements.

6.4.2. Subtitles and Closed Captioning

Technologies for subtitles are as follows:

- CEA-608/708 Digital Television (DTV) Closed Captioning [14]
- IMSC1 [61] conformant profiles of TTML, packaged as Segments conforming to MPEG-4, Part 30 [29], including subsets such as:
    - W3C TTML [16]
    - SMPTE Timed Text [17] (including image-based subtitles and closed captioning)
    - EBU-TT [20]
- 3GPP Timed Text [15]
- Web VTT [18]

For simple use cases, CEA-608/708 based signaling as defined in section 6.4.3 may be used.

For any other use cases, IMSC1 [61] should be used as defined in section 6.4.4. It is expected that most subset profiles of IMSC1 would be reasonably decodable.

TTML and WebVTT Media Segments shall be referenced by Representation elements in MPDs, downloaded, initialized, and synchronized for multimedia presentation the same as audio and video Segments.

> Note: DASH playback applications such as Web pages can download TTML or WebVTT text files, initialize renderers, and synchronize rendering and composition. This specification does not specify interoperable playback of these "sidecar" subtitle files in combination with a DASH audio visual presentation. However, section 6.4.5 provides guidelines on how to synchronize side car files at Period boundaries.

6.4.3. CEA-608/708 in SEI messages

6.4.3.1. Background

In order to provide the signaling of the presence of SEI-based data streams and closed captioning services on MPD level, descriptors on DASH level are defined. This section provides some background.

> Note: This method is compatible with draft SCTE specification DVS 1208 and therefore SCTE URNs are used for the descriptor `@schemeIdUri`. In an updated version of this document more details on the exact relation to the SCTE specification will be provided.

The presence of captions and their carriage within the SEI message of a video track is defined in ANSI/SCTE 128-1 2013 [43], section 8.1 Encoding and transport of caption, active format description (AFD) and bar data.

Based on this it is enabled that a video track carries SEI message that carry CEA-608/708 CC. The SEI message `payload_type=4` is used to indicates that Rec. ITU-T T.35 based SEI messages are in use.

In summary the following is included in ANSI/SCTE 128-1 2013 to signal CEA-608/708 CC:

- SEI `payloadType` is set to 4
- `itu_t_t35_country_code` – A fixed 8-bit field, the value of which shall be `0xB5`.
- `itu_t_35_provider_code` – A fixed 16-bit field registered by the ATSC. The value shall be `0x0031`.
- `user_identifier` – This is a 32 bit code that indicates the contents of the `user_structure()` and is `0x47413934` ("GA94").
- `user_structure()` – This is a variable length data structure `ATSC1_data()` defined in section 8.2 of ANSI/SCTE 128 2013-a.
- `user_data_type_code` is set to `0x03` for indicating captioning data in the `user_data_type_structure()`
- `user_data_type_structure()` is defined in section 8.2.2 of ANSI/SCTE 128-1 2013 for Closed Captioning and defines the details on how to encapsulate the captioning data.

The semantics of relevant Caption Service Metadata is provided in CEA-708 [14], section 4.5:

- the total number of caption services (1-16) present over some transport-specific period.
- For each service:

- The type of the service, i.e. being 608 or 708. According to CEA-708 [14], section 4.5, there shall be at most one CEA-608 data stream signaled. The CEA-608 datastream itself signals the individual CEA-608-E caption channels.
- When the type of the service is 708, then the following 708-related metadata should be conveyed:
  - SERVICE NUMBER: the service number as found on the 708 caption service block header (1-31). This field provides the linkage of the remaining metadata to a specific 708 caption service
  - LANGUAGE: the dominant language of the caption service, recommended to be encoded from ISO 639.2/B [45].
  - DISPLAY ASPECT RATIO {4:3, 16:9}: The display aspect ratio assumed by the caption authoring in formatting the caption windows and contents.
  - EASY READER: this metadata item, when present, indicates that the service contains text tailored to the needs of beginning readers.

6.4.3.2. MPD-based Signaling of SEI-based CEA-608/708 Closed Caption services

This subsection provides methods MPD-based Signaling of SEI-based CEA-608/708 Closed Caption services, i.e.

- The presence of one or several SEI-based closed caption services in a Representation.
- The signaling of the relevant Caption Service Metadata as defined in CEA-708 [14], section 4.5.

The descriptor mechanism in DASH is used for this purpose.

Signaling is provided by including **`Accessibility`** descriptors, one each for CEA 608 and CEA 708 and is described in sections 6.4.3.3 and 6.4.3.4, respectively. The **`Accessibility`** descriptor is included for the **`AdaptationSet`** and all included Representations shall provide equivalent captions.

The `@value` attribute of each descriptor can be either list of languages or a complete map of services (or CC channels, in CEA-608 terminology). Listing languages without service or channel information is strongly discouraged if more than one caption service is present.

These definitions are equivalent to SCTE 214-1 [56].

6.4.3.3. Signaling CEA-608 caption service metadata

The **`Accessibility`** descriptor shall be provided with `@schemeIdUri` set to `urn:scte:dash:cc:cea-608:2015`, and an optional `@value` attribute to describe the captions. If the `@value` attribute is not present, the Representation contains a CEA-608 based closed captioning service.

If present, the `@value` attribute shall contain a description of caption service(s) provided in the stream as a list of channel-language pairs. Alternatively, a simple list of language codes may be provided, but this is strongly discouraged as it will not provide sufficient information to map the language with the appropriate caption channel.

The `@value` syntax shall be as described in the ABNF below.

```
@value          = (channel *3 [";" channel]) / (language *3[";" language])
channel         = channel-number "=" language
channel-number  = CC1 | CC2 | CC3 | CC4
language        = 3ALPHA ; language code per ISO 639.2/B [45]
```

6.4.3.4. Signaling CEA-708 caption service metadata

DASH-IF IOPs do not provide any interoperability guidelines for CEA-708.

Note: Caption Service Metadata is provided in SCTE 214-1 [14], section 4.5.

6.4.3.5. Examples

Simple signaling of presence of CEA-608 based closed caption service (Note: Not signaling languages is a discouraged practice)

```
<Accessibility
    schemeIdUri="urn:scte:dash:cc:cea-608:2015"/>
```

Signaling of presence of CEA-608 closed caption service languages in English and German

```
<Accessibility
    schemeIdUri="urn:scte:dash:cc:cea-608:2015"
    value="eng;deu"/>
```

Signaling of presence of CEA-608 closed caption service in English and German, with channel assignments

```
<Accessibility
    schemeIdUri="urn:scte:dash:cc:cea-608:2015"
    value="CC1=eng;CC3=deu"/>
```

Signaling of presence of CEA-708 closed caption service in English and German

```
<Accessibility
    schemeIdUri="urn:scte:dash:cc:cea-708:2015"
    value="1=lang:eng;2=lang:deu"/>
```

Signaling of presence of CEA-708 closed caption service in English and easy reader English

```
<Accessibility
    schemeIdUri="urn:scte:dash:cc:cea-708:2015"
    value="1=lang:eng;2=lang:eng,war:1,er:1"/>
```

6.4.4. Timed Text (IMSC1)

W3C TTML [16] and its various profiles - W3C IMSC1 (text and image profiles) [61], SMPTE Timed Text [17], and EBU Timed Text [20] - provide a rich feature set for subtitles. Beyond basic subtitles and closed captioning, for example, graphics-based subtitles and closed captioning are also supported by IMSC1. Conversion of CEA-608 and CEA-708 into IMSC1 may be done according to SMPTE 2052-10 [27] and SMPTE-2052-11 [28], respectively. The Timed Text track shall conform to IMSC1 [61]. Note that by the choice of IMSC1 as the supported format at the client, other formats such as EBU TT [20] are also supported because they are subset profiles.

In the context of DASH, the following applies for text/subtitling:

- All graphics type samples shall be SAP type 1. The signalling of the different text/subtitling codecs for the codecs parameters is according to W3C TTML Profile Registry [62] and is documented in Table 21.
- Table 21 also provides information on ISO BMFF encapsulation.

Table 21 Subtitle MIME type and codecs parameter according to IANA and W3C registries

Codec	MIME type	Codecs Parameter @codecs	ISO BMFF Encapsulation
IMSC1 Timed Text [61] without encapsulation	`application/ttml+xml`(1,3)	See [62]	n/a
IMSC1 Timed Text [61] with ISO BMFF encapsulation	`application/mp4`	See [62]	ISO/IEC 14496-12 [7] ISO/IEC 14496-30 [29]
Notes: (1) DVB DASH only supports ISO BMFF encapsulated TT, but not XML-based.			

6.4.5. Guidelines for side-loaded TTML and WebVTT files

Side-loaded TTML or WebVTT subtitles or caption files can be used by some players including dash.js. Such files can be indicated in the manifest like:

```
<AdaptationSet contentType="text" mimeType="application/ttml+xml” lang="swe">
      <Role schemeIdUri="urn:mpeg:dash:role:2011" value="subtitle"/>
      <Representation id="xml_swe" bandwidth="1000">
         <BaseURL>sub_swe_short.xml</BaseURL>
      </Representation>
</AdaptationSet>
```

Only one file for the full period is permitted, practically limiting this use case to non-live content.

Such external files are assumed do have a timeline aligned with the Period, so that TTML time 00:00:00.000 corresponds to the start of the Period. The presentation time offset is expected to be not presented, and if present, expected to be ignored by the DASH client.

The same applies to side-loaded WebVTT files. In that case, the `@mimeType` is `text/vtt`.

If segmented subtitles are needed, such as for live sources, ISOBMFF-packaged TTML or WebVTT segments shall be used with timing according to [29]. In particular, this means that the TTML timing inside the segments is with respect to the media timeline. ".

6.4.6. Annotation of Subtitles

Subtitles should be annotated properly using descriptors available in ISO/IEC 23009-1, Specifically Role, Accessibility, Essential Property and Supplemental Property descriptors and the DASH role scheme may be used. Guidelines for annotation are for example provided in DVB DASH, section 7.1.2 or SCTE 214-1 [56], section 7.2.

7. Content Protection and Security

7.1. Introduction

DASH-IF IOPs do not intend to specify a full end-to-end DRM system. However DASH-IF IOP provides a framework for multiple DRMs to protect DASH content by adding instructions or *Protection System Specific*, proprietary information in predetermined locations in MPDs, or DASH content that is encrypted with Common Encryption as defined in ISO/IEC 23001-7 [30].

The Common Encryption ('`cenc`') protection scheme specifies encryption parameters that can be applied by a scrambling system and key mapping methods using a common key identifier (KID) to be used by different DRM systems such that the same encrypted version of a file can be combined with different DRM systems that can store proprietary information for licensing and key retrieval in the Protection System Specific Header Box ('`pssh`'), or in **`ContentProtection`** Descriptors in an MPD. The DRM scheme for each `pssh` is identified by a DRM specific `SystemID`.

The recommendations in this document reduce the encryption parameters and use of the encryption metadata to specific use cases for VOD and live content with key rotation.

The base technologies are introduced first followed by informative chapter on standardized elements. Additional Content Protection Constraints are then listed that are specific to conformance to DASH-264/AVC IOP.

7.2. HTTPS and DASH

The use of HTTP within DASH as a transport protocol, especially for retrieving media segments, inherently provides many advanced features such as caching and redirection. However, when not encrypted, data delivered over HTTP are in the clear, and there may not be any guarantee that the clear-text data are not being eavesdropped and tampered. If this is of concern, transport security in HTTP-based delivery may be achieved by using HTTP over TLS (HTTPS) as specified in RFC 2818. HTTPS is a protocol for secure communication over a computer network which is widely used on the Internet and also increasingly used for content streaming, mainly for the following purposes:

- authenticating the streaming server with which a streaming client is communicating,
- protecting the privacy of the exchanged data from eavesdropping by providing encryption of bidirectional communications between a client and a server, and
- ensuring integrity of the exchanged data against man-in-the-middle attacks against tampering with and/or forging the contents of the communication.

Many HTTP streaming technologies (as well as many web sites) are moving to HTTPS delivery, such as Netflix, Youtube, Facebook (see for example presentations at http://dashif.org/events/event-2015-08-20/). One of the main reasons is that with Cross Origin Resource Sharing (CORS) (http://www.w3.org/TR/cors/). As all web browsers and services (such as Facebook and Google) move to HTTPS and CORS does not allow that if the origin web page is HTTPS that you move to HTTP for resources delivered as links in the web page, the media also needs to be delivered through HTTPS.

MPEG-DASH explicitly permits the use of https as a scheme and hence, HTTP over TLS as a transport protocol. When using HTTPS in DASH, one can for instance specify that all media segments are delivered over HTTPS, by declaring that all the `<BaseURL>`'s are HTTPS based, as follow:

```
<BaseURL>https://cdn1.example.com/</BaseURL>
<BaseURL>https://cdn2.example.com/</BaseURL>
```

One can also use HTTPS for retrieving other types of data carried with an MPD that are HTTP-URL based, such as DRM licenses specified within the `<ContentProtection>` descriptor:

```
<ContentProtection schemeIdUri="http://example.net/052011/drm">
  <drm:License>https://MoviesSP.example.com/protect?license=kljklsdfiowek</drm:License>
</ContentProtection>
```

Because of the CORS recommendation of not moving from HTTPS requests to HTTP ones, whenever HTTPS is used for any portion of DASH content referenced within an MPD, it is recommended that HTTPS be used for the entirety of the DASH content referenced in the same MPD.

While using HTTPS in DASH provides good levels of trust and authenticity for data exchanged between DASH servers and clients connected over HTTPS, it should be pointed out that HTTPS only protects the transport link, but not the access to streaming content and the usage of streamed content. HTTPS itself does not imply user authentication and content authorization (or access control). This is especially the case that HTTPS provides no protection to any streamed content cached in a local buffer at a client for playback. For this reason, using HTTPS does not replace the need for content protection and digital rights management of streaming content.

There are also some impacts of using HTTPS in DASH that are going to be challenges in maintaining DASH advantages and end-user streaming experiences:

- CDN: Because of HTTPS encrypting the connection between a client and a server for each HTTPS session, it causes difficulties in caching encrypted DASH segment content on CDNs to reuse it for other sessions and other clients.
- Network: As use of HTTPS establishes a secure tunnel between a client and a server, it is impossible for underlying networks (or any entities on the data delivery path between the client and server) to manage and optimize encrypted data traffics over the networks, especially when the networks are mobile, based on inspecting and analyzing content of the traffics.
- Efficiency: Using HTTPS introduces "double encryption" when streaming content is already encrypted, for instance, using a content protection or DRM system. While delivering encrypted content over HTTPS may not increase content security, it does add extra encryption overhead and therefore latency in streaming encrypted content.

7.3. Base Technologies Summary

The normative standard that defines common encryption in combination with ISO BMFF is ISO/IEC 23001-7 [30]. It includes:

- Common ENCryption (CENC) of NAL structure video and other media data with AES-128 CTR mode
- Support for decryption of a single Representation by multiple DRM systems
- Key rotation (changing media keys over time)
- XML syntax for expressing a default KID attribute and pssh element in MPDs

The main DRM components are:

1. The **`ContentProtection`** descriptors in the MPD (see [4], 5.3.7.2-Table 9, 5.8.5.2 and [4] 5.8.4.1) that contains the URI for signaling of the use of Common Encryption or the specific DRM being used.
2. '`tenc`' parameters that specify encryption parameters and `default_KID` (see [30] 8.2). The '`tenc`' information is in the Initialization Segment. Any KIDs in Movie Fragment sample group description boxes override the 'tenc' parameter of the `default_KID`, as well as the 'not encrypted' parameter. Keys referenced by `KID` in sample group descriptions must be available when samples are available for decryption, and may be stored in a protection system specific header box (`'pssh'`) in each movie fragment box (`'moof'`). The `default_KID` information may also appear in the MPD (see [30] 11).
3. '`senc`' parameters that may store initialization vectors and subsample encryption ranges. The '`senc`' box is stored in each track fragment box ('`traf`') of an encrypted track (see [30] 7.1), and the stored parameters accessed using the sample auxiliary information offset box ('`saio`') and the sample auxiliary information size box ('`saiz`') (see [4] 8.7.8 and 8.7.9).
4. '`pssh`' license acquisition data or keys for each DRM in a format that is "Protection System Specific". '`pssh`' refers to the Protection System Specific Header box described in [30], 8.1.2. '`pssh'` boxes may be stored in Initialization or Media Segments (see [31] 8.1 and 8.2). It may also be present in a `cenc:pssh` element in the MPD (see [4] 5.8.4.1, [30] 11.2.1). `cenc:pssh` information in the MPD allows faster parsing, earlier access, identification of duplicate license requests, and addition of DRMs without content modification. '`pssh`' boxes in Initialization Segments are not recommended because they trigger a license request each time an Initialization Segment is processed in a Web browser for each Representation and bitrate switch.

   Note: The duplication of the `pssh` information in the Initialization Segment may cause difficulties in playback with HTML5 - EME based players. I.e. content will fail unless players build complex DRM specific license handling.
5. Key rotation is mainly used to allow changes in entitlement for continuous live content. It is used as defined in [30] with the following requirements:
   - Sample To Group Box (`'sbgp'`) and Sample Group Description Box (`'sgpd'`) of type '`seig'` are used to indicate the `KID` applied to each sample, and changes to `KIDs` over time (i.e. "key rotation"). (see [4] 8.9.4) `KIDs` referenced by sample groups must have the keys corresponding to those `KIDs` available when the samples in a Segment are available for decryption. Keys referenced by sample groups in a Segment may be stored in that Segment in Protection System Specific Header Boxes (`'pssh'`) stored in the Movie Fragment Box (`'moof'`). A version 1 `'pssh'` box may be used to list the `KID` values stored to enable removal of duplicate boxes if a file is defragmented.
   - Keys stored in Media Segment `'pssh'` boxes must be stored in the same DRM format for all users so that the same Media Segments can be shared by all users. User-specific information must be delivered "out of band", as in a "root" license

associated with the `default_KID`, which can be individualized for each DRM client, and control access to the shared 'pssh' information stored in Media Segments, e.g. by encrypting the keys stored in Segment 'pssh' boxes with a "root key" provided by the user-specific DRM root license. Common Encryption specifies 'pssh' to enable key storage in movie fragments/Segments; but it does not preclude other methods of key delivery that satisfy KID indexing and availability requirements.

- For details see Section 7.5.

7.4. ISO BMFF Support for Common Encryption and DRM

7.4.1. Box Hierarchy

The ISO Media Format carries content protection information in different locations. Their hierarchy is explained in the informational chapter below, followed by a reference on where these elements are standardized.

The following shows the box hierarchy and composition for relevant boxes, when using common encryption:

- `moov/pssh` (zero or one per system ID)
- `moov/trak/mdia/minf/stbl/stsd/sinf/schm` (one, if encrypted)
- `moov/trak/mdia/minf/stbl/stsd/sinf/schi/tenc` (one, if encrypted)
- `moof/traf/saiz` (one, if encrypted)
- `moof/traf/saio` (one, if encrypted)
- `moof/traf/senc` (one, if encrypted)

for key rotation

- `moof/traf/sbgp` (one per sample group)
- `moof/traf/sgpd 'seig'(sample group entry)` (one per sample group)
- `moof/pssh` (zero or one per system ID)

Graphical overviews of above structure for VOD content and live content are shown in Figure 21 and Figure 22 respectively.

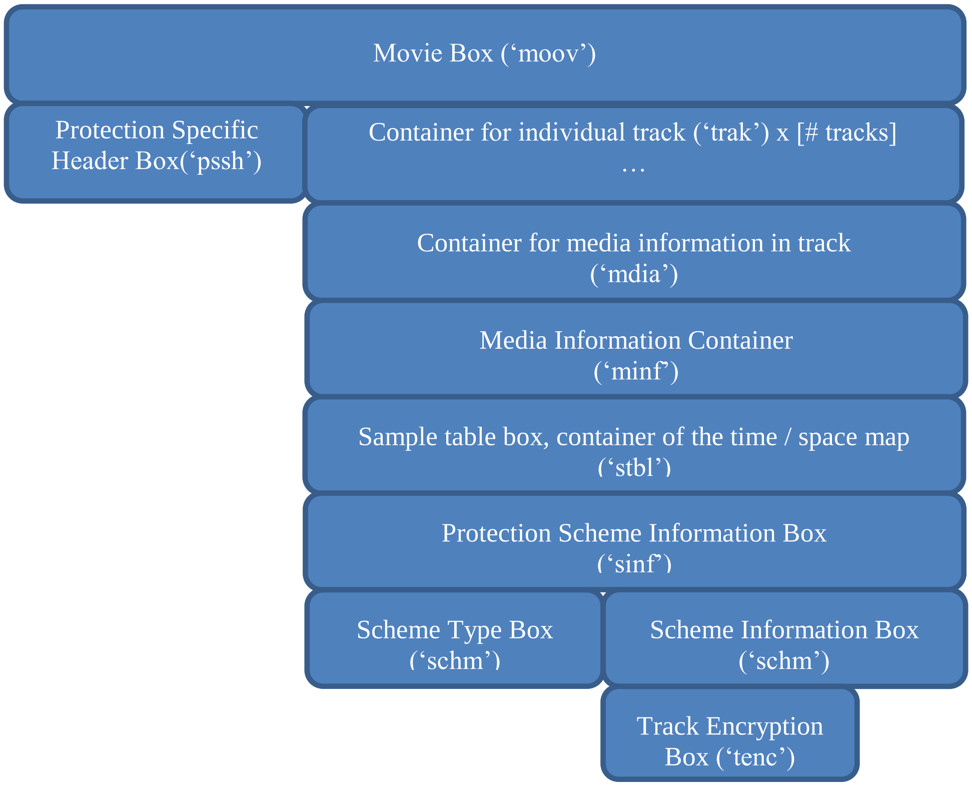

Figure 21: Visualization of box structure for single key content

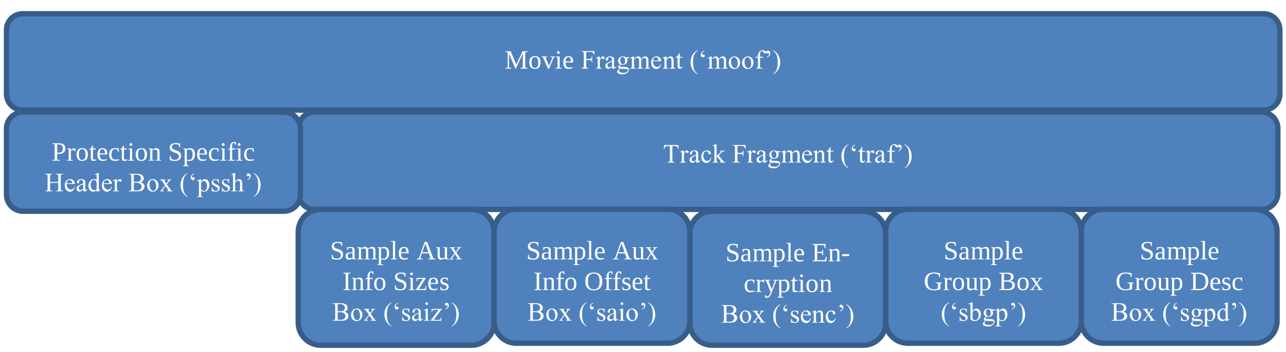

Figure 22: Visualization of box structure with key rotation

7.4.2. ISO BMFF Structure Overview

Table 22 provides pointers to relevant information in the specifications to understand the standard DRM components and if the main description is located in the ISO base media file format ([7]), or the Common Encryption specification ([30]).

Table 22 Boxes relevant for DRM systems

Box	**Full Name / *Usage***	**Reference**
`moof`	movie fragment header *One ‘moof’ box for each fragment, i.e. Media Segment/Subsegment.*	[7] 8.32 + [4]

moov	movie header, container for metadata *One 'moov' box per file.*	[7] 8.1
pssh	Protection System Specific Header Box *Contains DRM specific data. pssh box version 1 (specified in Common Encryption 2nd edition) contains a list of KIDs to allow removing duplicate 'pssh' boxes when defragmenting a file by comparing their KIDs*	[30] 8.1.1
saio	Sample Auxiliary Information Offsets Box *Contains the offset to the IVs & subsample encryption byte ranges.*	[7] 8.7.9
saiz	Sample Auxiliary Information Sizes Box *Contains the size of the IVs & subsample encryption byte ranges.*	[7] 8.7.8
senc	Sample Encryption Box *Contains Initialization Vectors; and subsample ranges for a Media Segment*	[30] 7.1
schi	Scheme Information Box *Container boxes used by that protection scheme type*.	[7] 8.12.6 + [30] 4
schm	Scheme Type Box *Contains the encryption scheme, identified by a 4 character code, e.g. 'cenc'*	[7], 8.12.5 + [30] 4
seig	Cenc Sample Encryption Information Video Group Entry *A sample description containing KIDs describing sample groups in this segment, for key rotation.*	[30] 6
sbgp	Sample to Group Box *lists a group of samples*	[7] +[30] 5
sgpd	Sample Group Description Box *Describes properties of a sample group*	[7] 8.9.3 + [30] 5
sinf	Protection Scheme Information Box *Signals that the stream is encrypted*	[7] 8.12.1 + [30] 4
stsd	Sample description table (codec type, initialization parameters, stream layout, etc.)	[7] 8.16
tenc	Track Encryption Box *Contains default encryption parameters for the entire track, e.g. default_KID*	[30] 8.2.1

7.5. Periodic Re-Authorization

7.5.1. Introduction

This section explains different options and tradeoffs to enable change in keys (aka key rotation), considering different use cases, application scenarios, content encoding variants and signaling requirements.

7.5.2. Use Cases and Requirements

The main use case in this context is to enable service changes at program boundaries, not to increase security of CENC by preventing e.g. key factoring or key redistribution. In order to clarify this application, the term *periodic re-authorization* is used instead of the term *key rotation*.

In addition, this is one of the ways to implement counting of active streams as they are periodically requesting keys from a license server.

The following use cases and requirements have been considered:

- Ability to force a client device to re-authorize to verify that it is still authorized for content consumption.
- Support for distribution models such as: Live content, PVR, PPV, VOD, SVOD, live to VOD, network DVR. This includes where live content is converted into another consumption license for e.g. catch up TV.
- Uninterrupted playback when keys are rotated.
    - Preventing of client storm: Requests from client should be distributed where possible to prevent spiking loads at isolated times.
    - Quick recovery: If the server or many client devices fail, the service should be able to resume quickly.
    - Player visibility into the key rotation signal
- Regional blackout: Device location may be taken into account to enable de-activation of content in a geographical area.
- Hybrid broadcast/unicast networks in which receivers operating in broadcast-only mode at least some of the time, i.e. unable to always download licenses on-demand through unicast.
- No required changes to the standard process and validity of MPDs.

7.5.3. Implementation Options

7.5.3.1. General

This section describes approaches for *periodic re-authorization;* recommended because they best cover the use cases and allow interoperable implementation. Other approaches are possible and may be considered by individual implementers.

One of those is explicit signaling using e.g. `esmg` messages, using a custom key rotation signal to indicate future KIDs.

To prevent the initial client storm to retrieve the first keys, before they are rotated, the initial `pssh` parameters SHOULD be included in the MPD as described in 7.4.1.

7.5.3.2. Period Boundaries

One possibility is to use a DASH Period as minimum key duration interval and existing MPD level signaling for KID.

This is a simple implementation and a possible alternative but has limitations in the flexibility:

- The signal does not allow for early warning and time to switch the encryption keys and context.
- The logic of the periods is decided by content creation not DRM. Boundaries may not be suited and period may be longer than desired key interval

7.5.3.3. Future Keys in `pssh`

This approach considers the protection system to be responsible to manage notification and key retrieval that prevents a client storm. The pssh information is used for signaling in a *content protected system* proprietary form. No additional signaling mechanism is created and the DRM is managing key rotation by providing extra information in the Protection System Specific Header Box (‘`pssh`’) (see [4]). To prevent a client storm on key change boundaries the following implementation options can be considered. They are listed for informational purpose and do not affect the guidelines on content formatting.

Current and future keys or access information and validity times are provided in a proprietary format in the `pssh` (see example in figure below). The client can chose a random time to use the access information to request licenses so that requests are distributed over time.

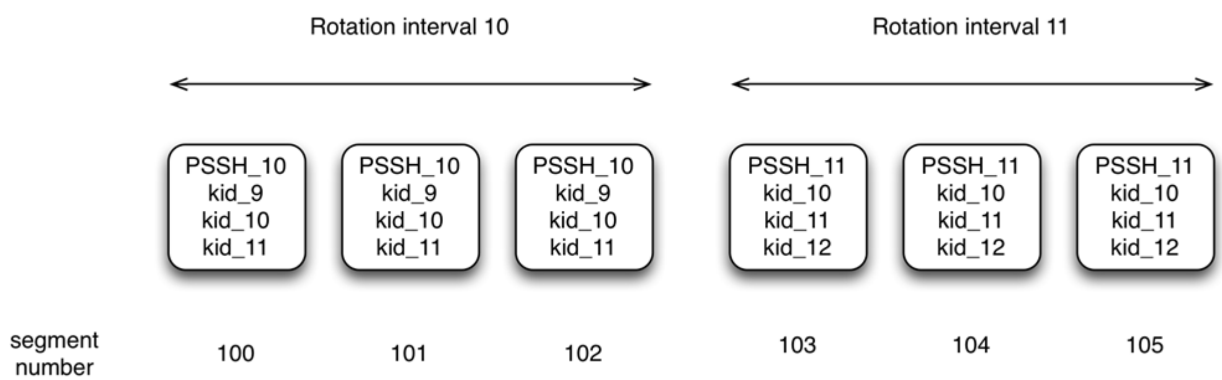

Figure 23: PSSH with version numbers and KIDs.

7.5.3.4. Key Hierarchy

The above approach also makes the protection system responsible to manage the key update and limits head end communication by using different types of licenses that established a hierarchy as follows:

- Entitlement Management License (EML) – A license a broadcaster can issue once to enforce some scope of content, such as a channel or library of shows (existing and future). It is cryptographically bound to one DRM domain associated with one user ID and, and enables access to ECLs and media keys associated with each show it authorizes.
- Entitlement Control License (ECL) – A license that contains a media key and can only be accessed by provisioned devices that have been authorized by installing the associated EML. ECLs may be delivered with the media in a broadcast distribution.

Changing media keys and ECLs per asset, forces re-authorization of each show by the DRM system which needs the media key.

When using any type of key hierarchy, the `default_KID` value in the **`ContentProtection`** element - which is also encoded into the `TrackEncryptionBox` (‘`tenc`’) - is the ID of the key which gives access to the content key(s). This is usually the key requested by the DRM client, and delivered in the EML.

7.6. MPD support for Encryption and DRM Signaling

7.6.1. Introduction

The MPD contains signaling of the content encryption and key management methods used to help the receiving client determine if it can possibly play back the content. The MPD elements to be used are the **`ContentProtection`** Descriptor elements. At least one **`ContentProtection`** Descriptor element SHALL be present in each **`AdaptationSet`** element describing encrypted content.

7.6.2. Use of the Content Protection Descriptor

7.6.2.1. `ContentProtection` Descriptor for *mp4protection* Scheme

A **`ContentProtection`** descriptor with the `@schemeIdUri` value equals to `"urn:mpeg:dash:mp4protection:2011"` signals that content is encrypted with the scheme indicated in the `@value` attribute. The file structure of content protection schemes is specified in [7], 5.8.5.2, and the `@value` = ‘`cenc`’ for the Common Encryption scheme, as specified in [30]. Although the **`ContentProtection`** Descriptor for UUID Scheme described below is usually used for license acquisition, the **`ContentProtection`** Descriptor with `@schemeIdUri="urn:mpeg:dash:mp4protection:2011"` and with `@cenc:default_KID` may be sufficient to acquire a license or identify a previously acquired license that can be used to decrypt the Adaptation Set. It may also be sufficient to identify encrypted content in the MPD when combined with license acquisition information stored in ‘`pssh`’ boxes in Initialization Segments.

A **`ContentProtection`** Descriptor for the mp4 Protection Scheme shall be used to identify the default `KID`, as specified by the ‘`tenc`‘ box, using the `@cenc:default_KID` attribute defined in [30], section 11.1. The value of the attribute is the `KID` expressed in UUID string notation.

```
<ContentProtection schemeIdUri="urn:mpeg:dash:mp4protection:2011"
       value="cenc" cenc:default_KID="34e5db32-8625-47cd-ba06-68fca0655a72"/>
```

When starting playback of any Adaptation Set, the client should interact with the DRM system to verify that the media key identified by the adaptation set’s default KID is available and should not assume that a media key is available for decrypting content unless so signaled by the DRM system.

When the `default_KID` is present on each Adaptation Set, it allows a player to determine if a new license needs to be acquired for each Adaptation Set by comparing their `default_KIDs` with each other, and with the `default_KIDs` of stored licenses. A player can simply compare these KID strings and determine what unique licenses are necessary without interpreting license information specific to each DRM system.

7.6.2.2. ContentProtection Descriptor for UUID Scheme

A UUID **ContentProtection** descriptor in the MPD may indicate the availability of a particular DRM scheme for license acquisition. An example is provided below:

```
<ContentProtection
    schemeIdUri="urn:uuid:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
    value="DRMNAME version"/>
```

The schemeIdUri uses a UUID URN with the UUID string equal to the registered SystemID for a particular DRM system. A list of known DRM SystemIDs can be found in the DASH identifier repository available here: `http://www.dashif.org/identifiers/protection`.

This is specified in [7], 5.8.5.2 and is referred to as "**ContentProtection** Descriptor for UUID Scheme" in the following.

7.6.2.3. Protection System Specific Header Box `cenc:pssh` element in MPD

A '`pssh`' box is defined by each DRM system for use with their registered `SystemID`, and the same box can be stored in the MPD within a **ContentProtection** Descriptor for UUID scheme using an extension element in the "`cenc:`" namespace. Examples are provided in [6] and in [30] sec. 11.2.

Carrying `cenc:default_KID` attribute and a `cenc:pssh` element in the MPD is useful to allow key identification, license evaluation, and license retrieval before live availability of initialization segments. This allows clients to spread license requests and avoid simultaneous requests from all viewers at the instant that an Initialization Segments containing license acquisition information in 'pssh' becomes available. With `cenc:default_KID` indicated in the mp4protection **ContentProtection** Descriptor on each Adaptation Set, clients can determine if that key and this presentation is not available to the viewer (e.g. without purchase or subscription), if the key is already downloaded, or which licenses the client SHOULD download before the `@availabilityStartTime` of the presentation based on the `default_KID` of each **AdaptationSet** element selected.

7.6.2.4 Use of W3C Clear Key with DASH

When using Clear Key [69] with MPEG DASH, Clear Key management availability is signaled in the MPD with a **ContentProtection** element that has the following format.

The Clear Key ContentProtection element attributes take the following values:

- The UUID `e2719d58-a985-b3c9-781a-b030af78d30e` is used for the `@schemeIdUri` attribute.
- The `@value` attribute is equal to the string "ClearKey1.0"

The following element MAY be added under the **ContentProtection** element:

- **Laurl** element that contains the URL for a Clear Key license server allowing to receive a Clear Key license in the format defined in [69] section 9.1.4. It has the attribute

@Lic_type that is a string describing the license type served by this license server. Possible value is “EME-1.0” when the license served by the Clear Key license server is in the format defined in [69] section 9.1.4.

The name space for the **Laurl** element is http://dashif.org/guidelines/clearKey

An example of a Clear Key **ContentProtection** element is as follows

```
<xs:schema xmlns:ck=http://dashif.org/guidelines/clearKey>

<ContentProtection
    schemeIdUri="urn:uuid:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
    value="ClearKey1.0">
       <ck:Laurl
           Lic_type="EME-1.0">
              https://clearKeyServer.foocompany.com</ck:Laurl>
</ContentProtection>
```

W3C also specifies the use of the SystemID=”1077efec-c0b2-4d02-ace3-3c1e52e2fb4b” in [70] section 4 to indicate that tracks are encrypted with Common Encryption [33], and list the KID key identifiers of keys used to encrypt the track in a version 1 ‘pssh’ box with that SystemID. However, the presence of this Common PSSH box does not indicate whether keys are managed by DRM systems or Clear Key management specified in this section. Browsers are expected to provide decryption in the case where Clear Key management is used, and a DRM system where a DRM key management system is used.

Therefore, clients SHALL NOT use the signalling of SystemID 1077efec-c0b2-4d02-ace3-3c1e52e2fb4b as an indication that the Clear Key mechanism is to be used.

W3C specifies that in order to activate the Clear Key mechanism, the client must provide Clear Key initialization data to the browser. The Clear Key initialization data consists of a listing of the default KIDs required to decrypt the content.

The MPD SHALL NOT contain Clear Key initialization data. Instead, clients SHALL construct Clear Key initialization data at runtime, based on the default KIDs signaled in the MPD using ContentProtection elements with the urn:mpeg:dash:mp4protection:2011 scheme.

When requesting a Clear Key license to the license server, it is recommended to use a secure connection as described in Section 7.2.

When used with a license type equal to “EME-1.0”:

- The GET request for the license includes in the body the JSON license request format defined in [69] section 9.1.3. The license request MAY also include additional authentication elements such as access token, device or user ID.
- The response from the license server includes in the body the Clear Key license in the format defined in [69] section 9.1.4 if the device is entitled to receive the Content Keys.

Clear Key licenses SHALL NOT be used to manage a key and KID that is also used by a DRM system. The use of an unprotected DRM key risks the security of DRM systems using that key, and violates the terms of use of most DRM systems.

7.7. Additional Content Protection Constraints

The following describes additional constraints for presentations to be conformant with DASH-264/AVC, for both MPD and ISO Media files.

7.7.1. ISO BMFF Content Protection Constraints

- There SHALL be identical values of `default_KID` in the Track Encryption Box (‘`tenc`’) of all Representation referenced by one Adaptation Set. Different Adaptation Sets may have equal or different values of `default_KID`.
- If a W3C Common ‘pssh’ box [69] is used with encrypted content, its list of KIDs SHALL contain only the default_KID from the ‘tenc’ box.
- ‘`pssh`’ boxes SHOULD NOT be present in Initialization Segments, and `cenc:pssh` elements in **`ContentProtection`** Descriptors used instead. If ‘`pssh`’ boxes are present in Initialization Segments, each Initialization Segment within one Adaptation Set SHALL contain an equivalent `pssh` box for each `SystemID`, i.e. license acquisition from any Representation is sufficient to allow switching between Representations within the Adaptation Set without acquiring a new license.

  Note: ‘`pssh`’ boxes in Initialization Segments may result in playback failure during browser playback when a license request is initiated each time an Initialization Segment is processed, such as the start of each protected Representation, each track selection, and each bitrate switch. This content requires DASH clients that can parse the ‘`pssh`’ box contents to determine the duplicate license requests and block them.

  A `cenc:pssh` element is parsed at most once per Adaptation Set by a client’s MPD parser, and the potential need for a new license request is identified by a new `cenc:default_KID` value. In this case, only the DASH client initiates license requests, and may do so per Period, if `cenc:default_KID` is a new value and the DRM system does not already have the key available for use.

7.7.2. MPD Content Protections Constraints

- For an encrypted Adaptation Set, **`ContentProtection`** Descriptors shall always be present in the **`AdaptationSet`** element, and apply to all contained Representations.
- A **`ContentProtection`** Descriptor for the mp4 Protection Scheme with the `@schemeIdUri` value of `"urn:mpeg:dash:mp4protection:2011"` and `@value='cenc'` shall be present in the **`AdaptationSet`** element if the contained Representations are encrypted.

  Note that this allows clients to recognize the Adaptation Set is encrypted with common encryption scheme without the need to understand any system specific UUID descriptors.

  The **`ContentProtection`** Descriptor for the mp4protection scheme should contain the optional attribute `@cenc:default_KID`. The ‘`tenc`’ box that specifies the encoded track encryption parameters shall be considered the source of truth for the default key ID value since it contains the `default_KID` field, and is present in the movie box, as specified in [30], section 8.2.1. The MPD cenc:default_KID attribute SHALL match the ‘tenc’ `default_KID`.

Note that this allows clients to identify the default KID from the MPD using a standard location and format, and makes it accessible to general purpose clients that don't understand the system specific information formats of all DRM schemes that might be signaled.

- The `cenc:pssh` element SHOULD be present in the **`ContentProtection`** Descriptor for each UUID Scheme. The base64 encoded contents of the element SHALL be equivalent to a `'pssh'` box including its header. The information in the `'pssh'` box SHOULD be sufficient to allow for license acquisition.

  Note: A player such as DASH.js hosted by a browser may pass the contents of this element through the Encrypted Media Extension (EME) API to the DRM system Content Decryption Module (CDM) with a SystemID equal to the Descriptor's UUID. This allows clients to acquire a license using only information in the MPD, prior to downloading Segments.

  Below is an example of the recommended format for a hypothetical acme DRM service:

```
<ContentProtection schemeIdUri="urn:uuid:d0ee2730-09b5-459f-8452-
200e52b37567"
        value="Acme DRM 2.0">
        <!-- base64 encoded 'pssh' box with SystemID matching the containing
ContentProtection Descriptor -->
        <cenc:pssh>
           YmFzZTY0IGVuY29kZWQgY29udGVudHMgb2YgkXB
           zc2iSIGJveCB3aXRoIHRoaXMgU3lzdGVtSUQ=
        </cenc:pssh>
</ContentProtection>
```

- The `@value` attribute of the **`ContentProtection`** Descriptor for UUID Scheme SHOULD contain the DRM system and version in a human readable form.

7.7.3. Other Content Protections Constraints

In the case where the '`pssh`' box element is present in the MPD and in the Initialization Segment, the '`pssh`' box element in the MPD SHALL take precedence, because the parameters in the MPD will be processed first, are easier to update, and can be assumed to be up to date at the time the MPD is fetched.

Recommended scheduling of License and key delivery:

- Request licenses on initial processing of an MPD if **`ContentProtection`** Descriptors or Initialization Segments are available with license acquisition information. This is intended to avoid a large number of synchronized license requests at **`MPD`**`@availabilityStartTime`.
- Prefetch licenses for a new Period in advance of its presentation time to allow license download and processing time, and prevent interruption of continuous decryption and playback. Advanced requests will also help prevent a large number of synchronized license requests during a live presentation at **`Period`**`@start` time.

7.7.4. Additional Constraints for Periodic Re-Authorization

- Key rotation should not occur within individual segments, as their duration is typically short enough to enable the intended use cases.

- Each Movie Fragment SHOULD contain one ‘pssh’ in each ‘moof’ box per SystemID that contains sufficient information for the DRM system with matching SystemID to obtain protected keys for this movie fragment, when combined with:
  - Information from ‘pssh’ in ‘moov’ or cenc:pssh in MPD.
  - KID associated with each sample from ‘seig’ sample group description box.
  - Sample to group boxes that list all the samples that use a particular KID.
- The KID should be observable by the player by reading the clear key_ids in PSSH definition v1.
- If the key is does not need to be retrieved, a pssh update may not result in a license request.
- If key_id cannot be observed, the player may perform binary comparison of pssh segments to understand updates.

7.7.5. Encryption of Different Representations

Representations contained in one Adaptation Set SHALL be protected by the same license for each protection system (“DRM”), and SHALL have the same value of ‘default_KID’ in their ‘tenc’ boxes in their Initialization Segments. This is to enable seamless switching within Adaptation Sets, which is generally not possible if a new DRM license needs to be authorized, client bound, generated, downloaded, and processed for each new Representation.

In the case of key rotation, if root licenses are used, the same requirement applies to the root licenses (one license per Adaptation Set for each DRM), and also means all Representations SHALL have the same value of ‘default_KID’ in their ‘tenc’ boxes in their Initialization Segments. The use of root and leaf licenses is optional and DRM specific, but leaf licenses are typically delivered in band to allow real time license acquisition, and do not require repeating client authentication, authorization, and rebuilding the security context with each key change in order to enable continuous playback without interruption cause be key acquisition or license processing.

In cases where SD and HD and UHD Representations are contained in one presentation, different license rights may be required for each quality level and may be sold separately. If different licenses are required for different quality levels, then it is necessary to create separate Adaptation Sets for each quality level, each with a different license and value of ‘default_KID’.

Representations that are equivalent resolution and bitrate but encrypted with different keys may be included in different Adaptation Sets. Seamless switching between UHD, HD and SD Representations is difficult because these quality levels typically use different decryption licenses and keys, use different DRM output rules (prohibit analog interfaces, require resolution down-scaling, require HDCP encryption on output, etc.), and use different decoding parameters for e.g. subsampling, codec, profile, bit depth, aspect ratios and color spaces.

If any Representation is encrypted in an Adaptation Set, then all must be encrypted using the same default_KID in the Track Encryption Box (‘tenc’) to avoid realtime changes to the DRM licenses and security context. KID values may change over time (“key rotation”) as specified in Common Encryption and a particular DRM system.

For all Representations within an Adaptation Set with @bitstreamSwitching=”false” (default), the following parameters shall apply.

- ‘tenc’ default_KID shall be equal for all Representations

7.7.6. Encryption of Multiple Periods

If a new license is needed and `cenc:default_KID` is to be changed, it SHALL be at the beginning of a Period. . A different file is indicated by a different `default_KID` signaled in the ‘`tenc`’ box in the Initialization Segment.

A file associated with a single license may be continued over multiple Periods by being referenced by multiple Representations over multiple Periods (for instance, a program interspersed with ad Periods). A client can recognize the same `cenc:default_KID` value and avoid having to download the same license again; but the DRM system may require a complete erase and rebuild of the security context, including all key material, samples in process, etc., between Periods with different licenses or no license (between protected and clear Periods).

7.7.7. DRM System Identification

The DRM system is signaled in the MPD and `'pssh'` boxes with a `SystemID`. A list of known DRMs can be found in the DASH identifier repository available here: http://www.dashif.org/identifiers/protection.

7.7.8. Protection of Media Presentations that Include SD, HD and UHD Adaptation Sets

Per DASH IF interop points, Representations with separate keys, licenses, and license policy are contained in different Adaptation Sets.

Adaptive bitrate switching can function automatically within an Adaptation Set without changing keys, licenses, robustness and output rules, etc.

A player may download licenses for multiple Adaptation Sets in a Group, and seamlessly switch between them if it is able. Seamless switching between Adaptation Sets is allowed, but not required. DASH may need to signal which Adaptation Sets are intended for seamless switching, i.e. have identical source content, same picture aspect ratio, same exact rescaled pixel registration, same sample description (e.g. ‘`avc3'`), same initialization behavior `(@bitstreamSwitching = true/false`), same Timescale and `@timescale`, and are mutually time-aligned.

The DASH-IF interop points are intended to make bitrate switching within an Adaptation Set simple and automatic, whether Representations are encrypted or not. Placement of Representations in different Adaptation Sets informs players that those Representations need to be initialized with different parameters, such as a different key and license. The full initialization process is repeated per Period. Adaptation Sets with `@bitstreamSwitching` = “`true`” only need to be initialized once per Period. Adaptation Sets with `@bitstreamSwitching = "false`” need to be partially re-initialized on each Representation switch (to change the SPS parameter sets referenced from NALs to those stored in in the containing track’s ‘avcC’), but most initialized parameters such as timescale, codec Profile/Level, display buffer size, colorspace, etc.; and licenses and the DRM system … do not need to be changed.

Fetching and resetting keys and licenses during adaptive switching requires processing Initialization Segments with different ‘`tenc'` default_KID and possibly ‘`pssh'` boxes. That may not be seamless, especially in browser playback where the decoders are only aware of player switching when an Initialization Segment flows through the MSE buffer and a `needKey()` event is raised via EME.

Note that switching between Adaptation Sets with different Media Profiles could be restricted by key and license policy, e.g. the user only purchased SD rights, the player only has analog output and HD content requires a protected digital output, UHD content requires hardware protected DRM, etc.

Implementations that seamlessly switch between Representations with different keys and policies generally require a standardized presentation ID or content ID system that associates multiple keys and licenses to that ID and presentation, then downloads only the keys/licenses authorized for that user and device (e.g. SD or HD+SD). The player must then install those licenses and use player logic to select only Representations in an Adaptation Set for which a license is installed and output controls, display configuration, etc. allow playback (e.g. only Representations keyed for an installed SD license). Players and license servers without this pre-configuration protocol and adaptive switching logic will encounter key/license requests in the process of adaptive switching, and may find output blocked by different license policies, user rights, etc.

7.7.9. Client Interactions with DRM Systems

The client interacts with one or more DRM systems during playback in order to control the decryption of content. Some of the most important interactions are:

1) Determining the availability of media keys.
2) Requesting the DRM system to acquire media keys.

In both of these interactions, the client and DRM system use the default_KID as an abstract mechanism to communicate information regarding the capability to decrypt adaptation sets that use a particular default_KID. A DRM system may also make use of other media keys in addition to the one signalled by default_KID (e.g. in key derivation or sample variant schemes) but this SHALL be transparent to the client, with only the default_KID being used in communications between the client and DRM system.

A client SHALL determine the required set of media keys based on the default KIDs signalled in the manifest for the adaptation sets selected for playback.

Upon determining that one or more required media keys signalled by default KIDs are not available the client SHOULD interact with the DRM system and request the missing media keys. The client MAY also request media keys that are known to be usable. Clients SHALL explicitly request all required media keys signaled by default KIDs and SHALL NOT assume that requesting one key from this set will implicitly make others available.

The client and/or DRM system MAY batch multiple key requests (and the respective responses) into a single transaction (for example, to reduce the chattiness of license acquisition traffic).

7.8. Workflow Overview

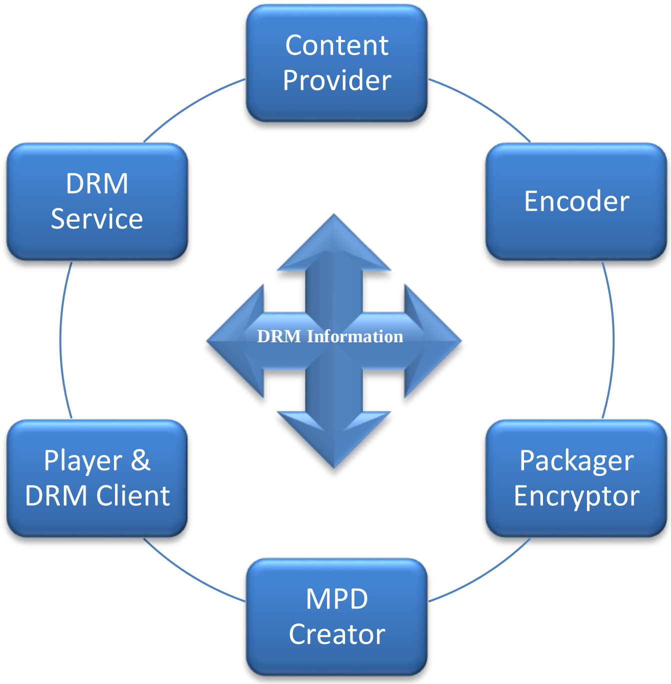

Figure 24 Logical Roles that Exchange DRM Information and Media

Figure 24 shows logical entities that may send or receive DRM information such as media keys, asset identifiers, licenses, and license acquisition information. A physical entity may combine multiple logical roles, and the point of origin for information, such as media keys and asset identifiers, can differ; so various information flows are possible. This is an informative example of how the roles are distributed to facilitate the description of workflow and use cases. Alternative roles and functions can be applied to create conformant content.

Description of Logical Roles:

Content Provider – A publisher who provides the rights and rules for delivering protected media, also possibly source media (mezzanine format, for transcoding), asset identifiers, key identifiers (KID), key values, encoding instructions, and content description metadata.

Encoder – A service provider who encodes Adaptation Sets in a specified media format, number of streams, range of bitrates and resolutions, seamless switching constraints, etc., possibly determined by the publisher. An asset identifier needs to be assigned to each encoded track in order to associate a key identifier, a Representation element in an MPD, a possible '`pssh`' box in the file header, and a DRM license separately downloaded.

Packager / Encryptor – A service provider who encrypts and packages media files, inserting `default_KID` in the file header '`tenc`' box, initialization vectors and subsample byte ranges in track fragments indexed by '`saio`' and '`saiz`' boxes, and possibly packages '`pssh`' boxes containing license acquisition information (from the DRM Provider) in the file header. Tracks that are partially encrypted or encrypted with multiple keys require sample to group boxes and sample group description boxes in each track fragment to associate different KIDs to groups of samples. The Packager could originate values for KIDs, media keys, encryption layout, etc., then send that information to other entities that need it, including the DRM Provider and Streamer, and probably the Content Provider. However, the Packager could receive that information from a different point of origin, such as the Content Provider or DRM Provider.

MPD Creator – The MPD Creator is assumed to create one or more types of DASH MPD, and provide indexing of Segments and/or '`sidx`' indexes for download so that players can byte range index Subsegments. The MPD must include descriptors for Common Encryption and DRM key management systems, and SHOULD include identification of the `default_KID` for each `AdaptationSet` element, and sufficient information in UUID `ContentProtection` Descriptor elements to acquire a DRM license. The `default_KID` is available from the Packager and any other role that created it, and the DRM specific information is available from the DRM Provider.

Player / DRM Client – Gets information from different sources: MPD, Media files and DRM License.

DRM Service – The DRM Provider creates licenses containing a protected media key that can only be decrypted by a trusted client.

The DRM Provider needs to know the `default_KID` and DRM SystemID and possibly other information like asset ID and player domain ID in order to create and download one or more licenses required for a Presentation on a particular device. Each DRM system has different license acquisition information, a slightly different license acquisition protocol, and a different license format with different playback rules, output rules, revocation and renewal system, etc. The DRM Provider typically must supply the Streamer and the Packager license acquisition information for each UUID `ContentProtection` Descriptor element or '`pssh`' box, respectively.

The DRM Service may also provide logic to manage key rotation, DRM domain management, revocation and renewal and other content protection related features.

Figure 25 shows a simple workflow with `pssh` information in the Initialization Segment for informational purpose.

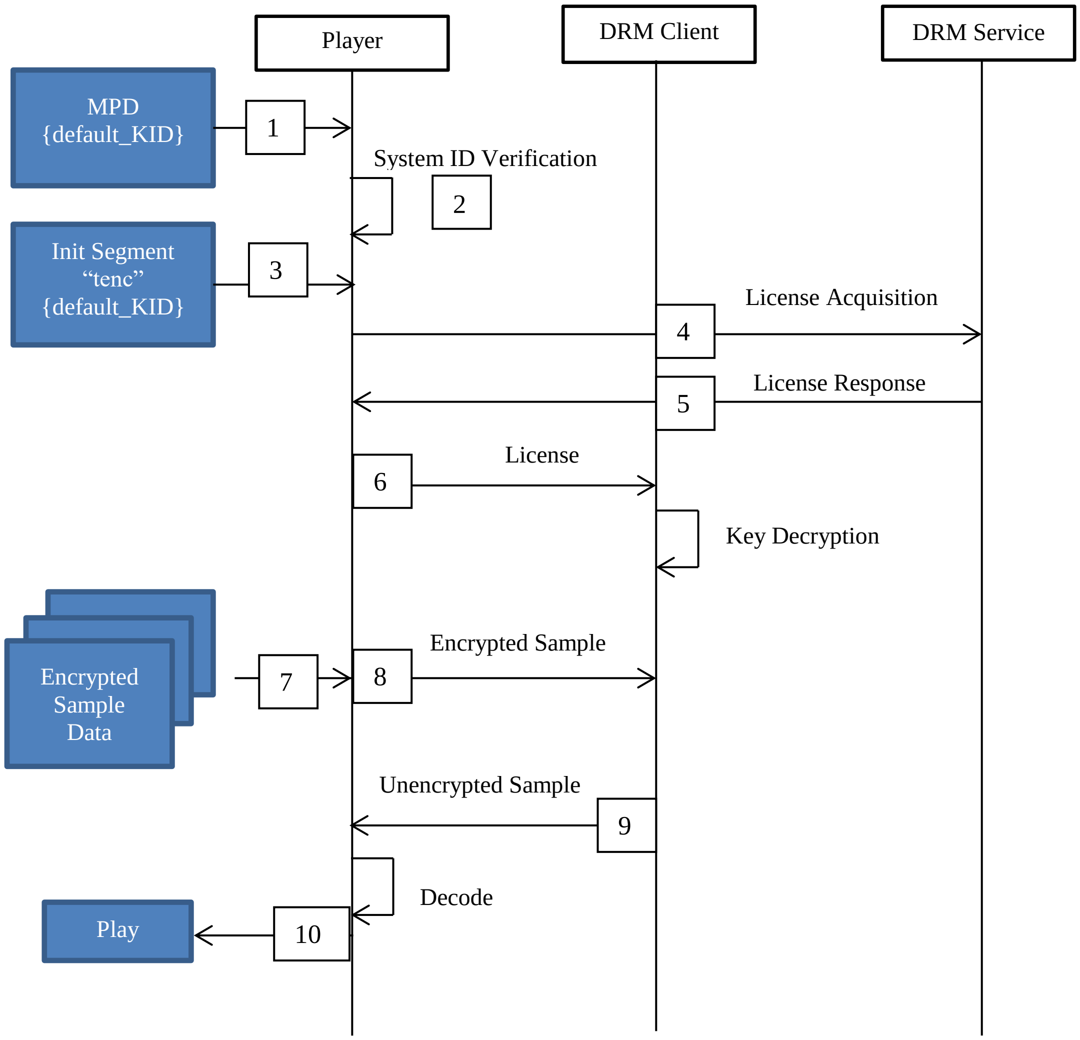

Figure 25 Example of Information flow for DRM license retrieval

[1] A MPD may include **`ContentProtection`** Descriptors to indicate that the '`cenc`' scheme is used to encrypt the referenced media, and to provide license acquisition information for one (or more) DRM system(s) with the indicated SystemID.

[2] The Player verifies if a specified DRM is supported using System ID value(s) from the MPD.

With unique KIDs, a license request using the `cenc:default_KID` attribute value is sufficient to identify a DRM license containing that key that will enable playback of the Components, Representations, Adaptation Sets, or Periods that the `ContentProtection` Descriptor element and `default_KID` describe.

[3] The TrackEncryptionBox ('`tenc`') contains default values for the `IsEncrypted`, `IV_size`, and `KID` for the entire track These values are used as the encryption parameters for the samples in this track unless over-ridden by the sample group description associated with a group of samples. The license acquisition information could be also present in '`pssh`' boxes in the initialization segment.

[4] Decryption Key acquisition can be performed either by the Player or the DRM Client.

[5] DRM License / Decryption Key response includes the required material for enabling access.

[6] DRM licenses/rights need not be stored in order to look up a key using KID values stored in the file and decrypt media samples using the encryption parameters stored in each track.

[7] The Player requesting encrypted sample data.

[8] The Player provides encrypted sample data to the DRM Client for decryption using the decryption key. How the DRM system locates the identified decryption key is left to the DRM.

[9] The Player received unencrypted sample data from the DRM Client.

8. DASH-IF Interoperability Points

8.1. Introduction

This version of the document defines Interoperability Points in this section. Earlier versions of this document, especially version 2 [2] defines legacy IOPs.

8.2. DASH-AVC/264 main

8.2.1. Introduction

The scope of the DASH-AVC/264 main interoperability point is basic support of high-quality video distribution over the top based on H.264/AVC up to 1080p. Both, live and on-demand services are supported.

The compliance to DASH-AVC/264 main may be signaled by a `@profiles` attribute with the value "`http://dashif.org/guidelines/dash264main`"

8.2.2. Definition

A DASH client conforms to the IOP by supporting at least the following features:

- All DASH-related features as defined in section 3 of this document.
- The requirements and guidelines in section 4.9.2 for simple live operation.
- The requirements and guidelines in section 5.6.1 for server-based ad insertion.
- H.264/MPEG AVC Progressive High Profile at level 4.0 as defined in section 6.2 together with all AVC-related requirements and recommendation in section 6.2.
- MPEG-4 HE-AAC v2 level 2 profile audio codec as defined in section 6.3. Dynamic Range Control is not expected to be supported.
- subtitle and closed captioning support
    - using SMPTE-TT as defined in section 6.4.2
        - For On-Demand single file download is sufficient.
        - For live services and/or if key rotation is to be supported, the encapsulation into ISO BMFF is necessary.
    - Using CEA-608/708 as defined in section 6.4.3.

- content protection based on common encryption and key rotation as defined in section 7. And specifically, the client supports MPD-based parsing and movie box based parsing of DRM related parameters for common encryption.

Content shall only be authored claiming conformance to this IOP if such a client can properly play the content. In addition, the content shall follow the mandatory aspects and should take into account the recommendations and guidelines for content authoring documented in section 3 (DASH features), section 4.9.2 (simple live operation), section 5.6.1 (server-based ad insertion), AVC-related issues in section 6.2, section 6.3 (audio), section 6.4.2 (SMPTE-TT), section 6.4.3 (CEA-608/708), and section 7 (Content Protection).

If content is offered claiming conformance to this IOP, the content author is encouraged to use the HTTP-URL construction as defined in [6], section 5.1.4.

8.3. DASH-AVC/264 high

8.3.1. Introduction

The scope of the DASH-AVC/264 interoperability point is support of high-quality video distribution over the top based on H.264/AVC up to 1080p. Both, live and on-demand services are supported as well as features for main live and advanced ad insertion.

The compliance to DASH-AVC/264 may be signaled by a `@profiles` attribute with the value "`http://dashif.org/guidelines/dash264high`"

8.3.2. Definition

A client that attempts to consume content generated conforming to this IOP shall support the following features:

- All features required for DASH-264/AVC main as defined in section 8.2.
- The client requirements and recommendations for the main live operation as defined in section 4.9.3.

Content shall only be authored claiming conformance to this IOP if such a client can properly play the content. In addition, the content shall follow the mandatory aspects and should take into account the recommendations and guidelines for content authoring documented in section 8.2 (DASH-264/AVC main), section 4.9.3 (main live operation), and section 5.6.2 (app-based ad insertion).

If content is offered claiming conformance to this IOP, the content author is encouraged to use the HTTP-URL construction as defined in [6], section 5.1.4.

8.4. DASH-IF IOP simple

8.4.1. Introduction

The scope of the DASH-IF IOP simple interoperability point is the basic support of efficient high-quality video distribution over the top with HD video up to 1080p including support for HEVC 8 bit.

The compliance to *DASH-IF IOP simple* may be signaled by a `@profiles` attribute with the value "`http://dashif.org/guidelines/dash-if-simple`"

8.4.2. Definition

A DASH client conforms to the IOP by supporting at least the following features:

- All DASH-related features as defined in section 3 of this document.
- The requirements and guidelines in section 4.9.2 for simple live operation.
- The requirements and guidelines in section 5.6.1 for server-based ad insertion.
- H.264/MPEG AVC Progressive High Profile at level 4.0 as defined in section 6.2 together with all AVC-related requirements and recommendation in section 6.2.
- H.265/MPEG-H HEVC Main Profile Main Tier at level 4.1 as defined in section 6.2 together with all HEVC-related requirements and recommendation in section 6.2.
- MPEG-4 HE-AAC v2 level 2 profile audio codec as defined in section 6.3. Dynamic Range Control is not expected to be supported.
- subtitle and closed captioning support
  - using SMPTE-TT as defined in section 6.4.2
    - For On-Demand single file download is sufficient.
    - For live services and/or if key rotation is to be supported, the encapsulation into ISO BMFF is necessary.
  - Using CEA-608/708 as defined in section 6.4.3.
- content protection based on common encryption and key rotation as defined in section 7. And specifically, the client supports MPD-based parsing and movie box based parsing of DRM related parameters for common encryption.

Content shall only be authored claiming conformance to this IOP if such a client can properly play the content. In addition, the content shall follow the mandatory aspects and should take into account the recommendations and guidelines for content authoring documented in section 3 (DASH features), section 4.9.2 (simple live operation), section 5.6.1 (server-based ad insertion), section 6.2 (video), section 6.3 (audio), section 6.4.2 (SMPTE-TT), section 6.4.3 (CEA-608/708), and section 7 (Content Protection).

If content is offered claiming conformance to this IOP, the content author is encouraged to use the HTTP-URL construction as defined in [6], section 5.1.4.

8.5. DASH-IF IOP Main

8.5.1. Introduction

For the support of broad set of use cases the DASH-IF IOP Main Interoperability Point is defined. In addition the features of DASH-264/AVC main as defined in section 8.2 the interoperability point requires DASH clients for real-time segment parsing and 10-bit HEVC.

The compliance to *DASH-IF IOP main* may be signalled by a `@profile` attribute with the value `"http://dashif.org/guidelines/dash-if-main"`

8.5.2. Definition

A client that attempts to consume content generated conforming to this IOP shall support the following features:

- All features required for DASH-264/AVC high as defined in section 8.3.

- H.265/MPEG-H HEVC Main Profile Main Tier at level 4.1 as defined in section 6.2 together with all HEVC-related requirements and recommendation in section 6.2.
- H.265/MPEG-H HEVC Main 10 Profile Main Tier at level 4.1 as defined in section 6.2 together with all HEVC-related requirements and recommendation in section 6.2.

Content shall only be authored claiming conformance to this IOP if such a client can properly play the content. In addition, the content shall follow the mandatory aspects and should take into account the recommendations and guidelines for content authoring documented in section 8.3 and HEVC-related issues in section 6.2.

If the content is authored such that it also conforms to DASH-264/AVC high as defined in section 8.3, then the profile identifier for DASH-264/AVC high shall be added as well. If the profile identifier is missing, the content may be considered as HEVC only content.

If content is offered claiming conformance to this IOP, the content author is encouraged to use the HTTP-URL construction as defined in [6], section 5.1.4.

9. Multi-Channel Audio Extensions

9.1. Scope

The Scope of the Multichannel Audio Extension is the support of audio with additional channels and codecs beyond the basic audio support as specified in the DASH-AVC/264 base, which is limited to Stereo HE-AAC. Multichannel audio is widely supported in all distribution channels today, including broadcast, optical disc, and digital delivery of audio, including wide support in adaptive streaming delivery.

It is expected that clients may choose which formats (codecs) they support.

9.2. Technologies

9.2.1. Dolby Multichannel Technologies

9.2.1.1. Overview

The considered technologies from Dolby for advanced audio support are:

- Enhanced AC-3 (Dolby Digital Plus) [35]
- Dolby TrueHD [36]
- AC-4 [63]

9.2.1.2. DASH-specific issues

In the context of DASH, the following applies:

- The signaling of the different audio codecs for the codecs parameters is documented in [35], [36] and [63] which also provides information on ISO BMFF encapsulation.
- For E-AC-3 and AC-4 the Audio Channel Configuration shall use the "`tag:dolby.com,2014:dash:audio_channel_configuration:2011`" as defined at http://dashif.org/identifiers/audio-source-data/.

Table 23 Dolby Technologies: Codec Parameters and ISO BMFF encapsulation

Codec	Codec Parameter	ISO BMFF Encapsulation	SAP type

Enhanced AC-3 [35]	`ec-3`	ETSI TS 102 366 Annex F [35]	1
Dolby TrueHD	`mlpa`	Dolby [36]	1
AC-4	`ac-4`	ETSI TS 103 190-1 Annex E [63]	1

9.2.2. DTS-HD

9.2.2.1. Overview

DTS-HD [37] comprises a number of profiles optimized for specific applications. More information about DTS-HD and the DTS-HD profiles can be found at www.dts.com.

9.2.2.2. DASH-specific issues

For all DTS formats SAP is always 1.

The signaling of the various DTS-HD profiles is documented in DTS 9302J81100 [34]. DTS 9302J81100 [34] also provides information on ISO BMFF encapsulation.

Additional information on constraints for seamless switching and signaling DTS audio tracks in the MPD is described in DTS specification 9302K62400 [39].

Table 24: DTS Codec Parameters and ISO BMFF encapsulation

Codec	Codec Parameter	ISO BMFF Encapsulation	SAP type
DTS Digital Surround	`dtsc`	DTS 9302J81100 [34]	1
DTS-HD High Resolution and DTS-HD Master Audio	`dtsh`		
DTS Express	`dtse`		
DTS-HD Lossless (no core)	`dtsl`		

9.2.3. MPEG Surround

9.2.3.1. Overview

MPEG Surround, as defined in ISO/IEC 23003-1:2007 [38], is a scheme for coding multichannel signals based on a down-mixed signal of the original multichannel signal, and associated spatial parameters. The down-mix shall be coded with MPEG-4 High Efficiency AAC v2 according to section 5.3.3.

MPEG Surround shall comply with level 4 of the Baseline MPEG Surround profile.

9.2.3.2. DASH-specific issues

In the context of DASH, the following applies for audio codecs

- The signaling of the different audio codecs for the codecs parameters is according to RFC6381 [10] is documented in Table 25. Table 25 also provides information on ISO BMFF encapsulation.

- The content is expected to be prepared according to the MPEG-DASH Implementation Guidelines [6] to make sure each (sub-)segment starts with a SAP of type 1.

Table 25 Codecs parameter according to RFC6381 [10] and ISO BMFF encapsulation for MPEG Surround codec

Codec	Codec Parameter	ISO BMFF Encapsulation	SAP type
MPEG Surround [38]	`mp4a.40.30`	ISO/IEC 14496-14 [8]	1

Note: Since MPEG Surround is based on a down-mix coded with AAC-LC and HE-AAC, for the above mentioned "Codec Parameters" the following is implied:

mp4a.40.30 = AOT 2 + AOT 5 + AOT 30

9.2.4. MPEG-4 High Efficiency AAC Profile v2, level 6

9.2.4.1. Overview

Support for multichannel content is available in the HE-AACv2 Profile, starting with level 4 for 5.1 and level 6 for 7.1. All MPEG-4 HE-AAC multichannel profiles are fully compatible with the DASH-AVC/264 baseline interoperability point for stereo audio, i.e. all multichannel decoders can decode DASH-IF IOPS stereo content.

9.2.4.2. DASH-specific issues

In the context of DASH, the following applies for the High Efficiency AAC v2 Profile

- The content shall be prepared according to the MPEG-DASH Implementation Guidelines [6] to make sure each (sub-)segment starts with a SAP of type 1.
- Signaling of profile levels is not supported in RFC 6381 but the channel configuration shall be signaled by means of the **`ChannelConfiguration`** element in the MPD.
- The signaling of MPEG-4 High Efficiency AAC v2 for the codecs parameters is according to RFC6381 [10] and is documented in Table 26. Table 26 also provides information on the ISO BMFF encapsulation.
- For all HE-AAC bitstreams, explicit backward-compatible signaling of SBR shall be used.
- The content should be prepared incorporating loudness and dynamic range information into the bitstream also considering DRC Presentation Mode in ISO/IEC 14496-3 [11], Amd. 4.
- Decoders shall support decoding of loudness and dynamic range related information, i.e. dynamic_range_info() and MPEG4_ancillary_data() in the bitstream.

Table 26 Codecs parameter according to RFC6381 [10] and ISO BMFF encapsulation

Codec	Codec Parameter	ISO BMFF Encapsulation	SAP type
MPEG-4 AAC Profile [11]	`mp4a.40.2`	ISO/IEC 14496-14 [12]	1
MPEG-4 HE-AAC Profile [11]	`mp4a.40.5`	ISO/IEC 14496-14 [12]	1
MPEG-4 HE-AAC v2 Profile [11]	`mp4a.40.29`	ISO/IEC 14496-14 [12]	1

Note: Since both, HE-AAC and HE-AACv2 are based on AAC-LC, for the above mentioned "Codec Parameters" the following is implied:

mp4a.40.5 = AOT 2 + AOT 5

9.2.5. MPEG-H 3D Audio

9.2.5.1. Overview

MPEG-H 3D Audio (MHA) including carriage in the ISO BMFF is specified in Amd. 2 of ISO/IEC 23008-3 [64].

9.2.5.2. DASH-specific issues

Storage of MHA elementary streams in the ISO BMFF shall be according to ISO/IEC 23008-3 [64] with the following constraints:

- An audio sample shall consist of a single MHA access unit
- The parameter values of `MHADecoderConfigurationRecord` and `MHASampleEntry` shall be consistent with the configuration in the MHA elementary stream
- `referenceChannelLayout` shall be '`ChannelConfiguration`' according to ISO/IEC 23001-8 [49].
- MHA bit streams shall contain an extension element of type ID_EXT_ELE_AUDIOPREROLL as specified in ISO/IEC 23008-3 sub- clause 5.5.5. [64]. At the start of each segment, the audio pre-roll extension payload of the first access unit in that Segment shall contain a valid configuration structure (AudioPreRoll.Config()) and should contain at least one pre-roll frame (AudioPreRoll.numPreRollFrames > 0).

Table 27 Codecs parameter and ISO BMFF encapsulation

Code	Codec Parameter	ISO BMFF Encapsulation	SAP Type
MPEG-H 3D Audio	`mha[1,2]`	ISO/IEC 23008-3:2015 Amd. 2	1

9.3. Client Implementation Guidelines

Independent of the codec, a client that supports one or more codecs of multichannel sound playback should exhibit the following characteristics:

- Playback multichannel sound correctly given the client operating environment. As an example, if the audio track delivers 5.1 multichannel sound, the client might perform one or more of the following: decode the multichannel signal on the device and output either 6ch PCM over HDMI, or pass that multichannel audio with no changes to external AVRs, or if the device is rendering to stereo outputs such as headphones, either correctly downmix that multi-channel audio to 2-channel sound, or select an alternate stereo adaptation set, or other appropriate choices.
- Adaptively and seamless switch between different bitrates as specified in the adaptation sets according to the playback clients logic. Seamless switching is defined as no perceptible interruption in the audio, and no loss of A/V sync. There is no expectation that a client can seamlessly switch between formats.

9.4. Extensions

9.4.1. General

9.4.1.1. Definitions

A *multichannel audio client* at least supports the following features:

- All DASH-related features as defined in section 3 of this document.
- content protection based on common encryption and key rotation as defined in section 7. And specifically, the client supports MPD-based parsing and movie box based parsing of DRM related parameters for common encryption.
- The client implementation guidelines in section 9.3.

9.4.1.2. Recommendations

If content is offered claiming conformance to any extension in this section, the content author is encouraged to use the HTTP-URL construction as defined in [6], section 5.1.4.

9.4.2. Dolby Extensions

9.4.2.1. Introduction

For the support of Dolby advanced audio support, three additional extensions are defined.

Conformance to *DASH-IF multichannel audio extension with Enhanced AC-3* (Dolby Digital Plus) [35] may be signaled by an `@profile` attribute with the value `"http://dashif.org/guidelines/dashif#ec-3"`.

Conformance to *DASH-IF multichannel extension with Dolby TrueHD* may be signaled by an `@profile` attribute with the value `"http://dashif.org/guidelines/dashif#mlpa"`.

Conformance to *DASH-IF multichannel extension with AC-4* may be signaled by an `@profile` attribute with the value `"http://dashif.org/guidelines/dashif#ac-4"`.

9.4.2.2. Supporters

These extensions are supported by the following DASH IF members: Dolby, DTS, Fraunhofer, BuyDRM, Sony.

9.4.2.3. Definition

Content may be authored claiming conformance to *DASH-IF multichannel audio extension with Enhanced AC-3*

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can properly play the content by supporting at least the following features
  - all multichannel audio client features as defined in section 9.4.1
  - Enhanced AC-3 (Dolby Digital Plus) [35] and the DASH-specific features defined in section 9.2.1.2

Content may be authored claiming conformance to *DASH-IF multichannel extension with Dolby TrueHD*

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can be properly play the content by supporting at least the following features

- all multichannel audio client features as defined in section 9.4.1
- Dolby TrueHD and the DASH-specific features defined in section 9.2.1.2

Content may be authored claiming conformance to *DASH-IF multichannel extension with AC-*

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can be properly play the content by supporting at least the following features
  - all multichannel audio client features as defined in section 9.4.1
  - AC-4 and the DASH-specific features defined in section 9.2.1.2

9.4.3. DTS-HD Interoperability Points

9.4.3.1. Introduction

For the support of DTS advanced audio support, four additional extensions are defined.

Conformance to *DASH-IF multichannel audio extension with DTS Digital Surround* may be signaled by a `@profile` attribute with value `"http://dashif.org/guidelines/dashif#dtsc"`.

Conformance to *DASH-IF multichannel audio extension with DTS-HD High Resolution and DTS-HD Master Audio* may be signaled by a `@profile` attribute with value `"http://dashif.org/guidelines/dashif#dtsh"`

Conformance to *DASH-IF multichannel audio extension with DTS Express* may be signaled by a `@profile` attribute with value `"http://dashif.org/guidelines/dashif#dtse"`

Conformance to *DASH-IF multichannel extension with DTS-HD Lossless (no core)* may be signaled by a `@profile` attribute with value `"http://dashif.org/guidelines/dashif#dtsl"`

9.4.3.2. Supporters

These extensions are supported by the following DASH IF members: Dolby, DTS, Fraunhofer, BuyDRM, Sony.

9.4.3.3. Definition

Content may be authored claiming conformance to *DASH-IF multichannel audio extension with DTS Digital Surround*

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can be properly play the content by supporting at least the following features
  - all multichannel audio client features as defined in section 9.4.1
  - DTS and the DASH-specific features defined in section 9.2.2.2

Content may be authored claiming conformance to *DASH-IF multichannel audio extension with DTS-HD High Resolution and DTS-HD Master Audio*

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can be properly play the content by supporting at least the following features
  - all multichannel audio client features as defined in section 9.4.1

- DTS-HD High Resolution and DTS-HD Master Audio and the DASH-specific features defined in section 9.2.2.2

Content may be authored claiming conformance to *DASH-IF multichannel audio extension with DTS Express*

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can be properly play the content by supporting at least the following features
  - all multichannel audio client features as defined in section 9.4.1
  - DTS-HD Express and the DASH-specific features defined in section 9.2.2.2

Content may be authored claiming conformance to *DASH-IF multichannel extension with DTS-HD Lossless (no core)*

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can be properly play the content by supporting at least the following features
  - all multichannel audio client features as defined in section 9.4.1
  - DTS-HD Lossless (no core) and the DASH-specific features defined in section 9.2.2.2

9.4.4. MPEG Surround Interoperability Points

9.4.4.1. Introduction

For the support of MPEG Surround advanced audio support the following extension is defined.

Conformance to *DASH-IF multichannel audio extension with MPEG Surround* according to ISO/IEC 23003-1:2007 [38] may be signaled by an `@profile` attribute with the value `"http://dashif.org/guidelines/dashif#mps"`.

9.4.4.2. Supporters

These extensions are supported by the following DASH IF members: Dolby, DTS, Fraunhofer, BuyDRM, Sony.

9.4.4.3. Definition

Content may be authored claiming conformance to *DASH-IF multichannel audio extension with MPEG Surround*

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can be properly play the content by supporting at least the following features
  - all multichannel audio client features as defined in section 9.4.1
  - ISO/IEC 23003-1:2007 and the DASH-specific features defined in section 9.2.3.2

9.4.5. MPEG HE-AAC Multichannel Interoperability Points

9.4.5.1. Introduction

Conformance to *DASH-IF multichannel audio extension with HE-AACv2 level 4* [11] may be signaled by an @profile attribute with the value "`http://dashif.org/guidelines/dashif#heaac-mc51`".

Conformance to *DASH-IF multichannel audio extension with HE-AACv2 level 6* [11] may be signaled by an @profile attribute with the value "`http://dashif.org/guidelines/dashif#heaac-mc71`".

9.4.5.2. Supporters

These extensions are supported by the following DASH IF members: Dolby, DTS, Fraunhofer, BuyDRM, Sony.

9.4.5.3. Definition

Content may be authored claiming conformance to *DASH-IF multichannel audio extension with HE-AACv2 level 4*

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can be properly play the content by supporting at least the following features
    - all multichannel audio client features as defined in section 9.4.1
    - HE-AACv2 level 4 [11] and the DASH-specific features defined in section 9.2.4.2

Content may be authored claiming conformance to *DASH-IF multichannel audio extension with HE-AACv2 level 6*

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can be properly play the content by supporting at least the following features
    - all multichannel audio client features as defined in section 9.4.1
    - HE-AACv2 level 6 [11] and the DASH-specific features defined in section 9.2.4.2

9.4.6. MPEG-H 3D Audio Interoperability Points

9.4.6.1. Introduction

Compliance to DASH-IF multichannel audio extension with MPEG-H 3D Audio [x] may be signaled by a `@profile` attribute with the value `http://dashif.org/guidelines/dashif#mha[1,2]`.

9.4.6.3. Definition

Content may be authored claiming conformance to DASH-IF multichannel audio extension with MPEG-H 3D Audio

- if the content is multichannel audio content as defined in section 9.4.1, and
- if a client can properly play the content by supporting at least the following features:
    - all multichannel audio client features as defined in section 9.4.1,
    - MHA and the DASH-specific features defined in section 9.4.6.

10. DASH-IF UHD Extensions

10.1. Introduction

This version of the document defines UHD Extensions in this section.

10.2. DASH-IF UHD HEVC 4k

10.2.1. Introduction

For the support of broad set of use cases the DASH-IF IOP HEVC 4k Extension is defined. UHD HEVC 4k video encoded with H.265/HEVC is an advanced distribution format for TV services that enables higher resolution experiences in an efficient manner.

In addition, the features of DASH-IF IOP Main as defined in section 8.5 and DASH-265/HEVC as defined in section 6.2.3, this extension adds the Main interoperability point to include 4k resolutions up to 60fps, and restricts the codec support to HEVC Main 10 Level 5.1.

The conformance to DASH-IF IOP HEVC 4k may be signaled by a `@profile` attribute with the value `http://dashif.org/guidelines/dash-if-uhd#hevc-4k`.

10.2.2. Elementary Stream Requirements

10.2.2.1. Constraints on Picture Formats

NAL Structured Video streams conforming to this Media Profile SHALL NOT exceed the following coded picture format constraints:

- Maximum encoded horizontal sample count of 3840 samples
- Maximum encoded vertical sample count of 2160 samples
- Maximum Frame Rate of 60000 / 1000.

Additional coded picture format constraints:

- The source video format shall be progressive.
- Representations in one Adaptation Set shall only differ on the following parameters: Bitrate, spatial resolution, frame rate
- The condition of the following SHALL NOT change throughout one HEVC video track:
    - `aspect_ratio_idc`
    - `cpb_cnt_minus1`
    - `bit_rate_scale`
    - `bit_rate_value_minus1`
    - `cpb_size_scale`
    - `cpb_size_value_minus1`
- The following fields should not change throughout an HEVC elementary stream:
    - `pic_width_in_luma_samples`
    - `pic_height_in_luma_samples`

  Note: A content provider should not change the parameters unless it is aware that the decoder and receiver can handle dynamic resolution switching, in particular switching from lower values to higher values. Clients should implement dynamic resolution switching based on DASH-IF IOP test vectors.
- YCbCr shall be used as the Chroma Format and 4:2:0 for color sub-sampling. The bit depth of the content shall be either 8 bit or 10 bit. The content shall be restricted to the HEVC video codec. See Section 10.2.2.2for details about HEVC encoding.

- The color primaries shall be ITU-R BT.709 [73].

10.2.2.2. Bitstream Requirements and Recommendations

A bitstream conforming to the H.265/HEVC 4k media profile shall comply with the Main10 Tier Main Profile Level 5.1 restrictions, as specified in Recommendation ITU-T H.265 / ISO/IEC 23008-2 [19].

UHD HEVC 4k bitstreams shall set `vui_parameters_present_flag` to 1 in the active Sequence Parameter Set, i.e. HEVC bitstreams shall contain a Video Usability Information syntax structure.

The sample aspect ratio information shall be signaled in the bitstream using the `aspect_ratio_idc` value in the Video Usability Information (see values of `aspect_ratio_idc` in Recommendation ITU-T H.265 / ISO/IEC 23008-2:2015 [19], table E-1). UHD HEVC 4k bitstreams shall represent square pixels indicated by `aspect_ratio_idc` shall be set to 1.

In addition to the provisions set forth in Recommendation ITU-T H.265 / ISO/IEC 23008-2:2015 [19], the following restrictions shall apply for the fields in the sequence parameter set:

- `vui_parameters_present_flag = 1`
- `sps_extension_flag = 0`
- `fixed_pic_rate_general_flag = 1`
- `general_interlaced_source_flag = 0`

In addition to the provisions set forth in Recommendation ITU-T H.265 / ISO/IEC 23008-2:2015[19], the following restrictions shall apply for the fields in the `profile_tier_level` syntax structure in the sequence parameter set:

- `general_tier_flag = 0`
- `general_profile_idc = 2`

UHD HEVC 4k bitstreams shall obey the limits in Recommendation ITU-T H.265 / ISO/IEC 23008-2:2015 [19], table A.1 and table A.2 associated to Level 5.1. `general_level_idc` shall be less than or equal to 153 (level 5.1).

It is recommended that bitstreams which are compliant with the Main or Main10 profile set `general_profile_compatibility_flag[1]` to `1`.

The chromaticity co-ordinates of the ideal display, opto-electronic transfer characteristic of the source picture and matrix coefficients used in deriving luminance and chrominance signals from the red, green and blue primaries shall be explicitly signaled in the encoded HEVC Bitstream by setting the appropriate values for each of the following 3 parameters in the VUI: `colour_primaries`, `transfer_characteristics`, and `matrix_coeffs`.

ITU-R BT.709 [73] colorimetry usage is signalled by setting `colour_primaries` to the value `1`, `transfer_characteristics` to the value `1` and `matrix_coeffs` to the value `1`.

The bitstream may contain SEI messages as permitted by the Recommendation ITU-T H.265 / ISO/IEC 23008-2:2015 [19]. Details on these SEI messages are specified in Recommendation ITU-T H.265 / ISO/IEC 23008-2 / Annex D.

10.2.2.3. Receiver Requirements

Receivers conforming to the HEVC 4k media profile shall support decoding and displaying H.265/HEVC 4k bitstreams as defined in clause 10.2.2.2.

No additional processing requirements are defined, for example processing of SEI messages is out of scope.

10.2.3. Mapping to DASH

10.2.3.1. Media Profile Identifier

If all Representations in an Adaptation Set conforms to the elementary stream constraints for the Media Profile as defined in clause 10.2.2.3 and the Adaptation Set conforms to the MPD signaling according to clause 10.2.3.2 and 10.2.3.4, and the Representations conform to the file format constraints in clause 10.2.3.3, then the `@profiles` parameter in the Adaptation Set may signal conformance to this operation point by using "`http://dashif.org/guidelines/dash-if-uhd#hevc-4k`".

10.2.3.2. MPD Signaling

The MPD shall conform to DASH-IF HEVC Main IOP with the additional constraints defined in clause 10.2.3.4. The `@codecs` parameter shall not exceed and should be set to either "`hvc1.2.4.L153.B0`" or "`hev1.2.4.L153.B0`".

10.2.3.3. File Format Requirements

Representations used in the context of the specification shall conform to the ISO BMFF Segment format [7], [9] with the following further requirements:

- The value of the `duration` field in the Movie Header Box ('`mvhd`') shall be set to a value of '0'
- The Track Header Box ('`tkhd`') shall obey the following constraints:
  - The value of the `duration` field shall be set to '0'.
  - The `width` and `height` fields for a visual track shall specify the track's visual presentation size as fixed-point 16.16 values expressed in on a uniformly sampled grid (commonly called square pixels)
- The Media Header Box ('`mdhd`') shall obey the following constraints:
  - The value of the `duration` field shall be set to '0'.
- The Video Media Header ('`vmhd`') shall obey the following constraints:
  - The value of the `version` field shall be set to '0'.
  - The value of the `graphicsmode` field shall be set to '0'.
  - The value of the `opcolor` field shall be set to {'0', '0', '0'}.

- The Sample Description Box (‘`stsd`’) shall obey the following constraints:
    - A visual sample entry shall be used.
    - The box shall include a NAL Structured Video Parameter Set
    - the maximum width and height values shall correspond to the maximum cropped horizontal and vertical sample counts indicated in any Sequence Parameter Set in the track
    - It shall contain a Decoder Configuration Record which signals the Profile, Level, and other parameters in the video track.
- The `entry_count` field of the Sample-to-Chunk Box (‘`stsc`’) shall be set to ‘0’.
- Both the `sample_size` and `sample_count` fields of the Sample Size Box (‘`stsz`’) box shall be set to zero (‘0’). The sample_count field of the Sample Size Box (‘`stz2`’) box shall be set to zero (‘0’). The actual sample size information can be found in the Track Fragment Run Box (‘`trun`’) for the track.

Note: this is because the Movie Box (‘`moov`’) contains no media samples.

- The `entry_count` field of the Chunk Offset Box (‘`stco`’) shall be set to ‘0’.
- Any Segment Index Box (‘`sidx`’), if present, shall obey the additional constraints:
    - The `timescale` field shall have the same value as the timescale field in the Media Header Box (‘`mdhd`’) within the same track; and
    - the `reference_ID` field shall be set to the `track_ID` of the ISO Media track as defined in the Track Header Box (‘`tkhd`’).
- For HEVCSampleEntry (‘`hev1`’) NAL Structured Video tracks, the '`first_sample_flags`' shall signal the picture type of the first sample in each movie fragment as specified below.
    - `sample_is_non_sync_sample`=0: If the first sample is a sync sample.
    - `sample_is_non_sync_sample`=1: If the first sample is not a sync sample.
    - `sample_depends_on`=2: If the first sample is an I-frame.
- The Colour Information Box should be present. If present, it shall signal the transfer characteristics of the elementary stream.
- The sample timing shall obey the frame rate requirements.

10.2.3.4. Adaptation Set Constraints

For a video Adaptation Set, the following constraints apply, which are identical to the constraints as specified in clause 3.2.10:

- The `@codecs` parameter shall be present on Adaptation Set level and shall signal the maximum required capability to decode any Representation in the Adaptation Set.
- The `@profiles` parameter may be present to signal the constraints for the Adaptation Set
- The attributes `@maxWidth` and `@maxHeight` shall be present. They are expected be used to signal the source content format. This means that they may exceed the actual largest size of any coded Representation in one Adaptation Set.
- The `@width` and `@height` shall be signalled for each Representation (possibly defaulted on Adaptation Set level) and shall match the values of the maximum width and height in the Sample Description box of the contained Representation.
- The attributes `@minWidth` and `@minHeight` should not be present. If present, they may be smaller than the smallest @width or smallest @height in the Adaptation Set.
- The maximum frame rate may be signalled on Adaptation Set using the `@maxFrameRate` attribute.
- The `@frameRate` should be signalled for each Representation (possibly defaulted on Adaptation Set level).

In addition to the above referenced constraints, this profile specifies the following additional contraints:

- The Color Space in use may be signalled. If signalled,
    - an Essential or Supplemental Descriptor shall be used to signal the value by setting the `@schemeIdUri` attribute to `urn:mpeg:mpegB:cicp:MatrixCoefficients` as defined ISO/IEC 23001-8 [49] and the `@value` attribute according to Table 4 of ISO/IEC 23001-8 [49]. The values shall match the values set in the VUI.
    - The signalling shall be on Adaptation Set level, i.e all Representations in one Adaptation Set are required to have the same Chroma Format.
- The Color Primaries and Transfer Function may be signalled. If signalled,
    - Essential or Supplemental Descriptors shall be used to signal the value by setting the `@schemeIdUri` attribute to `urn:mpeg:mpegB:cicp: ColourPrimaries` and `urn:mpeg:mpegB:cicp:TransferCharacteristics`, respectively, as defined ISO/IEC 23001-8 [49] and the `@value` attribute according to the “Colour primaries” Table and the “Transfer characteristics” Table of ISO/IEC 23001-8 [49], respectively. The values shall match the values set in the VUI.
    - The signalling shall be on Adaptation Point Set level only, i.e all Representations in one Adaptation Set are required to have the same Color Primaries and Transfer Function.

10.2.4. **Compatibility Aspects**

This specification is designed such that content that is authored in conformance to this IOP is expected to conform to the media profile defined by DVB DASH in ETSI TS 103 285 [42] and following the *3GPP H.265/HEVC UHD Operation Point* in section 5.6 of 3GPP TS26.116 [77]. However, in contrast to DVB and 3GPP, only BT.709 may be used and not BT.2020.

In addition, clients conforming to this extension should be capable to play content authored as conform to the media profile defined by DVB DASH in ETSI TS 103 285 [42] and following the *3GPP H.265/HEVC UHD Operation Point* in section 5.6 of 3GPP TS26.116 [77], if BT.709 colour space is used.

10.3. **DASH-IF IOP HEVC HDR PQ10**

10.3.1. **Introduction**

For the support of broad set of use cases addressing higher dynamic range (HDR) and wide colour gamut (WCG), the DASH-IF IOP HEVC HDR Perceptual Quantization (PQ) 10 Extension is defined. This interoperability point allows for additional UHD features including Wide Color Gamut, High Dynamic Range and a new electro-optical transfer curve. These features are in addition to the existing features described in the DASH-IF UHD 4k interoperability point, except that that this profile is designed for HDR, and requires the use of SMPTE ST 2084 [71] and Rec. BT-2020 [74] colour space. Note that this is identical to Rec. BT-2100 [80], PQ transfer function, Y'C'BC'R color difference formats, with 10 bit signal representation and narrow range.

Note that this Extension does not require the use of the maximum values, such as 60fps or 4K resolution. The content author may offer lower spatial and temporal resolutions and may use the regular DASH signalling to indicate the actual format of the source and rendering format. Typical cases may be to use HDR together with an HD 1080p signal. Note also that Adaptation Set Switching as defined in section 3.8 may be used to separate different spatial resolutions in different Adaptation Sets to address different capabilities, but still permit the use of lower resolutions for service continuity of higher resolutions.

The compliance to DASH-IF IOP HEVC HDR PQ10 may be signaled by a `@profile` attribute with the value `http://dashif.org/guidelines/dash-if-uhd#hevc-hdr-pq10`.

10.3.2. **Elementary Stream Requirements**

10.3.2.1. Introduction

The same requirements as for UHD HEVC 4k as documented in section 10.2 hold, expect for the changes as detailed below.

The changes in the HEVC HDR PQ10 profile that extend it beyond the HEVC 4K profile include:

- NAL Structured Video Streams conforming to this interoperability point SHALL be encoded using the REC-2020 color parameters as defined in [74]. Clients shall be able to correctly decode content that is encoded using that color space.
- NAL Structured Video Streams conforming to this interoperability point SHALL be encoded using the SMPTE ST 2084 electro-optic transfer function as defined in [71]. Clients shall be able to correctly decode content that is encoded using that electro-optic transfer function. Note that one cannot author a single piece of content that is compliant

with both this profile and HEVC 4k profile. However, the content may be offered in one MPD in two different Adaptation Sets.

Optional metadata may be present in form SEI messages defined in ITU-T H.265 /ISO/IEC 23008-2:2015 [19].

10.3.2.2. Bitstream Requirements and Recommendations

A bitstream conforming to the HEVC HDR PQ10 media profile shall comply with the Main Tier Main10 Profile Level 5.1 restrictions, as specified in Recommendation ITU-T H.265 / ISO/IEC 23008-2 [19].

In addition the requirements in section 10.2.2.2 apply, except that this profile requires the use of Recommendation ITU-R BT.2020 [74] non-constant luminance colorimetry and SMPTE ST 2084 [71].

SMPTE ST 2084 [71] usage shall be signaled by setting `colour_primaries` to the value `9`, `transfer_characteristics` to the value `16` and `matrix_coeffs` to the value `9`.

The bitstream may contain SEI messages as permitted by the Recommendation ITU-T H.265 / ISO/IEC 23008-2:2015 [19]. Details on these SEI messages are specified in Recommendation ITU-T H.265 / ISO/IEC 23008-2 / Annex D. SEI message may for example support adaptation of the decoded video signals to different display capabilities or more detailed content description, in particular those specified in Recommendation ITU-T H.265 / ISO/IEC 23008-2 / Annex D in relation to HDR. Other SEI Messages defined in ITU-T H.265 / ISO/IEC 23008-2 / Annex D may be present as well.

10.3.2.3. Receiver Requirements

Receivers conforming to the HEVC HDR PQ10 media profile shall support decoding and displaying HEVC HDR PQ10 bitstreams as defined in section 10.3.2.2.

No additional processing requirements are defined, for example processing of SEI messages is out of scope.

10.3.3. Mapping to DASH

10.3.3.1. Media Profile Identifier

If all Representations in an Adaptation Set conforms to the elementary stream constraints for the Media Profile as defined in clause 10.3.3.2 and the Adaptation Set conforms to the MPD signalling according to clause 10.3.3.2 and 10.3.3.4, and the Representations conform to the file format constraints in clause 10.3.3.3, then the `@profiles` parameter in the Adaptation Set may signal conformance to this operation point by using "`http://dashif.org/guidelines/dash-if-uhd#hevc-hdr-pq10`".

10.3.3.2. MPD Signaling

The MPD shall conform to DASH-IF HEVC Main IOP as defined with the additional constraints defined in clause 10.3.3.4. The `@codecs` parameter shall not exceed and should be set to either `"hvc1.2.4.L153.B0"` or `"hev1.2.4.L153.B0"`.

10.3.3.3. File Format Requirements

The file format requirements as defined in clause 10.2.3.3 shall apply.

10.3.3.4. Adaptation Set Constraints

The same requirements as defined in clause 10.2.3.4 shall apply.

10.3.4. Compatibility Aspects

Content authored according to this extensions is expected to be interoperable with the HDR10 profile defined in the DECE CFF Content Specification v2.2 [78], although it should be noted that the DECE CFF profile may have additional constraints, such as bitrate restrictions and required metadata.

Content authored according to this extensions is expected to be interoperable with the PQ10 package defined in the UHD Forum Guidelines phase A [79].

10.4. DASH-IF IOP UHD Dual-Stream (Dolby Vision1)

10.4.1. Introduction

For the support of broad set of backward compatible use cases the DASH-IF IOP Dual-Stream (Dolby Vision) Interoperability Point is defined. Backward Compatible refers to a simple method for one delivery format to satisfy both an HDR client and an SDR client. This Interoperability Point allows for two interlocked video streams, as described in the clause 10.4.2 below (restrictions to Enhancement Layers and Annex D 1.1). These two layers are known as the Base and Enhancement layers, where the Base Layer fully conforms to previous non-UHD or UHD DASH-IF Interoperability point. The EL provides additional information, which combined with the BL in a composition process produces a UHD output signal, including Wide Color Gamut and High Dynamic Range signal at the client.

The compliance to *DASH-IF IOP Dual-Stream (Dolby Vision)* may be signaled by a `@profile` attribute on the Enhancement Layer with the value `http://dashif.org/guidelines/dash-if-uhd#dvduallayer`

10.4.2. Definition

10.4.2.1. General

The dual-stream solution includes two video streams, known as the Base Layer and the Enhancement Layer. The high-level overview of the dual-stream process is shown in Figure 26 Overview of Dual-stream System.

[1] Note: This This extension is designed to be compatible with the “Dolby Vision Media Profile Definition” in DECE “Common File Format & Media Formats Specification” Version 2.2. The name of the DASH-IF extension is inherited from the DECE document in order to indicate the compatibility with this DECE Media Profile.

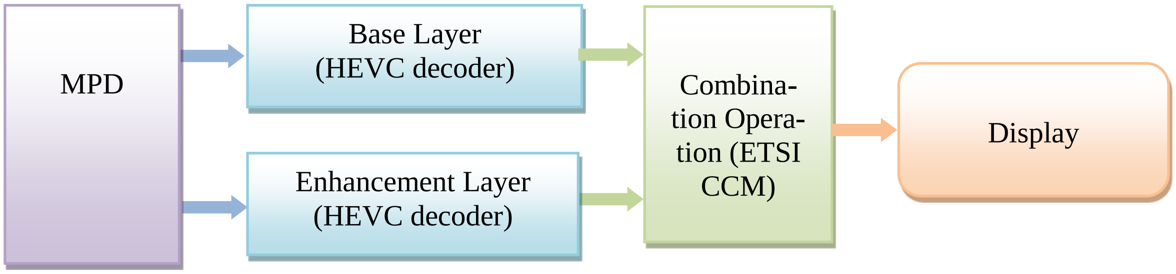

Figure 26 Overview of Dual-stream System

The MPD includes at least two Adaptation Sets as described below, including a Base Layer Adaptation Set and an Enhancement Layer Adaptation Set.

The Base Layer shall conform to the requirements of one of the following Interoperability Points: the DASH-IF IOP Main Interoperability Point, the DASH-IF IOP UHD 4k Interoperability point *or* the DASH-IF IOP UHD HDR10 Interoperability point. Any client that is able to play DASH-IF IOP Main content, DASH-IF IOP UHD 4k content, or DASH-IF IOP UHD HDR10 content as appropriate will be able to play the content from the Base Layer track as determined by the client capabilities. To be clear, the Base Layer is 100% conforming, with no changes or additional information, to the profile definition. A client that plays content conforming to the Base Layer profile will be able to play the Base Layer content with no modification and no knowledge of the Enhancement Layer or and Dolby Vision specific information. See Annex E, Sample MPD, for an example dual-layer MPD.

In addition, The Enhancement Layer shall conform to H.265/HEVC Main10 Profile Main Tier as defined in Recommendation ITU-T H.265 / ISO/IEC 23008-2, Level 5.1 or lower The Enhancement Layer shall conform to the following additional requirements:

- The Frame Rate is identical to the Base Layer video track.
- The EL DPB (Decoded Picture Buffer) shall support the same number of maximum frames as the maximum number of frames supported by the BL's DPB.
- If the Base layer sample contains an IDR picture, the Enhancement Layer sample must have an IDR picture at the same presentation time.
- Fragment durations and Presentation times are identical to the Base Layer video track. To clarify, "Presentation times are identical" means that for each picture at one layer, there shall be a picture at the other layer with the same presentation time.
- Each Enhancement Layer track has one and only one associated Base Layer video track (i.e. tracks are paired 1:1).

The client - may either play the Base Layer alone, in which case it complies with the requirements of those interoperability points, or the client plays the Base Layer and Enhancement Layer together, decoding both layers and combining them to produce a 12 bit enhanced HDR signal which conforms to REC.2020 color parameters and SMPTE-2084 electro-optical transfer function. The details of this combination operation are detailed in ETSI Specification "Compound Content Management" [85].

Content shall only be authored claiming conformance to this IOP if a client can properly play the content through the method of combining the Base Layer and Enhancement layers to produce an enhanced HDR output. Note that clients who conform to the profile associated with the Base Layer alone may play the Base Layer alone, with no information (and no knowledge) of the Enhancement Layer. In addition, the content shall follow the mandatory aspects and should take into account the recommendations and guidelines for content authoring documented in sections 8 and 10and HEVC-related issues in this section.

The dual-stream delivery of Dolby Vision asset uses two tracks; the Base Layer is written into one track according to the profile of the Base Layer, and the Enhancement Layer exists in a second track, per the [TBD Reference on integration, 12] specification and the details in Annex C and Annex D. In particular, details about required mp4 Boxes and sample entries are detailed in Annex C, "Dolby Vision Streams Within the ISO Base Media File Format"
The Enhancement Layer is identified by an additional parameter, `@dependencyId`, which identifies the Base layer which is the match for the Enhancement Layer as described in clause 10.4.2.3.

10.4.2.2. Bitstream Requirements for Enhancement Layer

The sample aspect ratio information shall be signaled in the bitstream using the `aspect_ratio_idc` value in the Video Usability Information (see values of `aspect_ratio_idc` in Recommendation ITU-T H.265 / ISO/IEC 23008-2:2013 [19], table E-1).

In addition to the provisions set forth in Recommendation ITU-T H.265 / ISO/IEC 23008-2:2013 [19], the following restrictions shall apply for the fields in the sequence parameter set:

- `bit_depth_luma_minus8` shall be set to "2".
- `aspect_ratio_idc` shall be set to "1".
- `general_interlaced_source_flag` shall be set to "0".

10.4.2.3. Supplemental Enhancement Information for Enhancement Layer

10.4.2.3.1. General

In addition to the requirements imposed in clause 10.4.2.2, the following additional specifications shall apply to the Enhancement Layer encoding:
HEVC Enhancement Layer Bitstreams shall contain the following SEI messages:

- User data registered by Recommendation ITU-T T.35[IT35] SEI message containing the message CM_data() (named composing metadata SEI message), as described in clause 10.4.2.3.3.
- User data registered by Recommendation ITU-T T.35[IT35] SEI message containing the message DM_data() (named display management SEI Message), as described in clause 10.4.2.3.4.
- Mastering display colour volume SEI message as specified in Recommendation ITU-T H.265 / ISO/IEC 23008-2 Annex D with the following constraints:
    - A valid number shall be set for the following syntax elements: `display_primaries_x[c]`, `display_primaries_y[c]`, `white_point_x`, `white_point_y`, `max_display_mastering_luminance` and `min_display_mastering_luminance`.

10.4.2.3.2. SEI User Data Syntax

CM_data() messages and DM_data() messages are carried in the enhancement layer video elementary stream as Supplemental Enhancement Information in HEVC's "User data registered by Recommendation ITU-T T.35 SEI message" syntactic element. The syntax of the composing metadata SEI message and the display management SEI message is defined in Table 28.

Table 28: Compound Content Management SEI message: HEVC (prefix SEI NAL unit with nal_unit_type = 39, payloadType=4)

user_data_registered_itu_t_t35(payloadSize) {	**Descriptor**
itu_t_t35_country_code	b(8)
itu_t_t35_provider_code	u(16)
user_identifier	u(32)
user_data_type_code	u(8)
user_data_type_structure()	
}	

itu_t_t35_country_code: This 8-bit field shall have the value 0xB5.

itu_t_t35_provider_code: This 16-bit field shall have the value 0x0031.

user_identifier: This 32-bit code shall have the value 0x47413934 ("GA94").

user_data_type_code: An 8-bit value that indentifies the type of user data to follow in the user_data_type_structure(). The values are defined in Table 29.

Table 29: UserID: user identifier

user_data_type_code	**user_data_type_structure()**
0x00 to 0x07	Reserved
0x08	CM_data()

0x09	DM_data()
0x0A to 0xFF	Reserved

user_data_type_structure(): This is a variable length set of data defined by the value of user_data_type_code and table C.1 (DM_data()) or table D.1 (CM_data()).

10.4.2.3.3. **Composing Metadata SEI Message**

The composing metadata SEI message is a "user data registered by Recommendation ITU-T T.35 SEI message" containing a CM_data() message, as specified in Annex F.

HEVC Enhancement Layer Bitstreams shall contain composing metadata SEI messages with the following constraints:

- It shall be sent for every access unit of the HEVC Enhancement Layer Bitstream.
- Bitstreams shall conform to ETSI Profile 1 as defined in Annex A of [85] and the value of the syntax element `ccm_profile` shall be set to "1".
- The value of the syntax element `ccm_level` shall be set to "0".
- The value of `BL_bit_depth_minus8` shall be set to "2".
- The value of `EL_bit_depth_minus8` shall be set to to "2".
- The value of the syntax element `hdr_bit_depth_minus8` shall be set to "2" or "4".

10.4.2.3.4. **Display Management SEI Message**

The display management SEI message is a "user data registered by Recommendation ITU-T T.35 SEI message" containing a DM_data() message, as specified in Annex C.

HEVC Enhancement Layer Bitstreams shall contain display management SEI messages with the following constraints:

- It shall be sent for every access unit of the HEVC Enhancement Layer Bitstream.
- `app_identifier` shall be set equal to "1".
- `app_version` shall be set equal to "1".
- The number of extension blocks with `ext_block_level` equal to "1" shall be constrained to be equal to "1".
- The number of extension blocks with `ext_block_level` equal to "2" shall be constrained to be less than or equal to "16".
- The number of extension blocks with `ext_block_level` equal to "5" shall be constrained to be equal to "0" or "1".

10.4.3. Mapping to DASH

10.4.3.1. Media Profile Identifier

If all Representations in an Adaptation Set conforms to the elementary stream constraints for the Media Profile as defined in clause 10.4.2.1 and the Adaptation Set conforms to the MPD signaling according to clause 10.4.3.2 and 10.4.3.3, and the Representations conform to the file format constraints in clause 10.4.3.4, then

- the `@profiles` parameter in the Adaptation Set may signal conformance to this operation point by using "http://dashif.org/guidelines/dash-if-uhd#dvdual-layer on the Enhancement Layer (the Base Layer uses the normal signaling of the layer as defined in the profile of the Base Layer).

10.4.3.2. MPD Signaling

The MPD shall conform to DASH-IF HEVC Main IOP as defined with the additional constraints defined in clause 10.4.2.

10.4.3.3. Codec Parameter Signaling

When the Dual-Stream Dolby Vision asset is delivered as two files, the Enhancement Layer is identified by an additional parameter, `@dependencyId`, which identifies the Base Layer that is the match for the Enhancement Layer. The Base Layer Representation element must have an `@id` attribute, and the `@dependencyId` attribute on the Enhancement Layer Representation shall refer to that `@id`, to indicate to a client that these two representations are linked. Note that in this case, the `@codecs` attribute for the Base Layer will have only the Base Layer codec. In this example, the Base Layer `@codecs` might be:

```
codecs="hvc1.1.0.L120.00"
```

And the Enhancement Layer `@codecs` would be:

```
codecs="dvhe.dtr.uhd30"
```

For both the Base Layer and the Enhacncement Layer, HEVC decoders are used in accordance with the `@codecs` signaling on each layer. The syntax and semantics of the `@codecs` signaling on the enhancement layer is detailed in Annex D. The output of the decoders are combined by the method detailed in ETSI Specification "Compound Content Management" [85].

10.4.3.4. File Format Requirements

Content shall only be authored claiming conformance to this IOP if a client can properly play the content. In addition, the content shall follow the mandatory aspects and should take into account the recommendations and guidelines for content authoring documented in clause 8 and 10 and HEVC-related issues in clause 6.2.

11. DASH-IF VP9 Extensions

11.1. Introduction

VP9 [86] is an alternative video codec is which may be used for SD, HD, and UHD spatial resolutions, as well as HDR10 and HDR12 bit depths (HDR + WCG); and frame rates of 24fps and higher. This codec provides significant bandwidth savings at equivalent qualities with respect to AVC/H.264. While not meant to replace AVC and HEVC, DASH presentations may include additional VP9 representations for playback on clients which support it.

11.2. DASH-Specific Aspects for VP9 Video

11.2.1. General

For the integration in the context of DASH, the following applies for VP9:

- The encapsulation of VP9 video data in ISO BMFF is defined in the VP Codec ISO-BMFF Binding specification [87]. Clients shall support both sample entries containing `'vp09'` and `'vpcC'` boxes, i.e. inband storage for `VPCodecConfigurationBox + VPCodecConfigurationRecord`.
- For delivery to consumer devices, only VP9 profile 0 (4:2:0 chroma subsampling and 8-bit pixel depth), and profile 1 (4.2.0 chroma subsampling and 10- or 12-bit pixel depths) shall be used.
- Stream Access Points shall coincide with the beginning of key frames (uncompressed header field `frame_type` = 0) as defined in the VP9 Bitstream Specification [86] section 7.2. Only type-1 SAPs are supported. Fragmentation and segmentation shall occur only at these points.
- Codec and codec configuration signaling in the MPD shall occur using the codec string defined in the VP Codec Binding Specification [87], DASH Application section.
- Encryption shall be signaled by the same mechanisms as defined in Common Encryption for ISO-BMFF Containers 3rd edition. Subsample encryption is required as per the VP Codec ISO Media File Format Binding spec [87].

11.2.2. Bitstream Switching

For VP9 video streams, if the `@bitstreamSwitching` flag is set to true, then the following additional constraints shall apply:

- Edit lists shall not be used to synchronize video to audio and presentation timelines.
- Video Media Segments shall set the first presented sample's composition time equal to the first decoded sample's decode time, which equals the `baseMediaDecodeTime` in the Track Fragment Decode Time Box (‘`tfdt`’).
    - Note: This requires the use of negative composition offsets in a v1 Track Run Box (‘`trun`’) for video samples, otherwise video sample reordering will result in a delay of video relative to audio.
- The `@presentationTimeOffset` attribute shall be sufficient to align audio, video, subtitle, and presentation timelines at presentation a Period's presentation start time. Any edit lists present in Initialization Segments shall be ignored. It is strongly recommended that the Presentation Time Offset at the start of each Period coincide with the first frame of a Segment to improve decoding continuity at the start of Periods.
- All representations within the Adaptation set shall have the same picture aspect ratio.
- All VP9 decoders are required to support dynamic video resolutions, however pixel bit-depths may not vary within an adaptation set. Because of this the encoding Profile must remain constant, but the Level may vary.
- All Representations within a video Adaptation Set shall include an Initialization Segment containing an ‘`vpcC`’ Box containing a Decoder Configuration Record with the highest, , Level, vertical and horizontal resolutions of any Media Segment in the Representation.

- The **`AdaptationSet`**`@codecs` attribute shall be present and contain the maximum level of any Representation contained in the Adaptation Set.
- The **`Representation`**`@codecs` attribute may be present and in that case shall contain the maximum level of any Segment in the Representation.

11.3. DASH-IF VP9 Extension IOPs

11.3.1. DASH-IF VP9-HD

The scope of the DASH-IF VP9-HD extension interoperability point is basic support of high-quality video distribution over the top based on VP9 up to 1080p with 8-bit pixel depth and up to 30fps. Both, live and on-demand services are supported.
The compliance to DASH-VP9 main may be signaled by a `@profiles` attribute with the value `"http://dashif.org/guidelines/dashif#vp9"`

A DASH client conforms to this extension IOP by supporting at least the following features:

- All DASH-related features as defined in clause 3 of this document.
- The requirements and guidelines in section 4.9.2 for simple live operation.
- The requirements and guidelines in section 5.6.1 for server-based ad insertion.
- Content protection based on common encryption and key rotation as defined in section 7. And specifically, the client supports MPD-based parsing parameters for common encryption.
- All VP9 DASH IF IOP requirements in clause 11.2.
- VP9 Profile 0 up to level 4.1.

11.3.2. DASH-IF VP9-UHD

The scope of the DASH-IF VP9-UHD extension interoperability point is basic support of high-quality video distribution over the top based on VP9 up to 2160p with 8-bit pixel depth and up to 60fps. Both, live and on-demand services are supported.
The compliance to DASH-VP9 main may be signaled by a `@profiles` attribute with the value `"http://dashif.org/guidelines/dash-if-uhd#vp9"`

A DASH client conforms to this extension IOP by supporting at least the following features:

- All features supported by DASH-IF VP9-HD defined in clause 11.3.1.
- VP9 Profile 0 up to level 5.1.

11.3.3. DASH-IF VP9-HDR

The scope of the DASH-IF VP9-HDR extension interoperability point is basic support of high-quality video distribution over the top based on VP9 up to 2160p with 10-bit pixel depth and up to 60fps. Both, live and on-demand services are supported.

The compliance to DASH-VP9 main may be signaled by a `@profiles` attribute with the value `http://dashif.org/guidelines/dashif#vp9-hdr (up to HD/1080p resolution), or http://dashif.org/guidelines/dash-if-uhd#vp9-hdr (up to 4K resolution).`

A DASH client conforms to this extension IOP by supporting at least the following features:

- All features supported by DASH-IF VP9-UHD defined in clauses 11.3.2.
- VP9 profile 2 up to level 5.1.
- Pixel depths of 10 bits.

Annex A Examples for Profile Signalling

Example 1

In this case DASH-IF IOP content is offered, but in addition a non-conforming Adaptation Set is added.

Here is an example for an MPD:

- **MPD**@profiles="urn:mpeg:dash:profile:isoff-on-demand:2011, http://dashif.org/guidelines/dash264"
    - **AdaptationSet**@profiles="urn:mpeg:dash:profile:isoff-on-demand:2011, http://dashif.org/guidelines/dash264"
    - **AdaptationSet**@profiles ="http://dashif.org/guidelines/dash264"
    - **AdaptationSet**@profiles ="urn:mpeg:dash:profile:isoff-on-demand:2011"

Pruning process for IOP `http://dashif.org/guidelines/dash264` results in

- **MPD**@profiles ="http://dashif.org/guidelines/dash264"
    - **AdaptationSet**@profiles ="http://dashif.org/guidelines/dash264"
    - **AdaptationSet**@profiles ="http://dashif.org/guidelines/dash264"

It is now required that the pruned MPD conforms to DASH-IF IOP.

Example 2

In this case DASH-IF IOP content is offered, but in addition a non-conforming Adaptation Set is added and one DASH-IF Example Extension Adaptation Set is added with the virtual IOP signal `http://dashif.org/guidelines/dashif#extension-example`.

Here is an example for an MPD:

- **MPD**@profiles ="urn:mpeg:dash:profile:isoff-on-demand:2011, http://dashif.org/guidelines/dash264, http://dashif.org/guidelines/dashif#extension-example"
    - @id = 1, **AdaptationSet**@profiles ="urn:mpeg:dash:profile:isoff-on-demand:2011, http://dashif.org/guidelines/dash264"
    - @id = 2, **AdaptationSet**@profiles ="http://dashif.org/guidelines/dash264"
    - @id = 3, **AdaptationSet**@profiles ="urn:mpeg:dash:profile:isoff-on-demand:2011, http://dashif.org/guidelines/dashif#extension-example"

Pruning process for profile `http://dashif.org/guidelines/dash264` results in

- MPD@profiles="http://dashif.org/guidelines/dash264"
    - @id = 1, **AdaptationSet**@profiles="http://dashif.org/guidelines/dash264"
    - @id = 2, **AdaptationSet**@profiles="http://dashif.org/guidelines/dash264"

It is now required that the pruned MPD conforms to DASH-IF IOP.

Pruning process for profile `http://dashif.org/guidelines/dashif#extension-example` results in

- **MPD**@profiles="http://dashif.org/guidelines/dash264"
    - @id = 3, **AdaptationSet**@profiles="http://dashif.org/guidelines/dashif# extension-example"

It is now required that the pruned MPD conforms to DASH-IF Example Extension Adaptation Set.

Annex B Live Services - Use Cases and Architecture

B.1 Baseline Use cases

B.1.1 Use Case 1: Live Content Offered as On-Demand

In this case content that was distributed as live is offered in a separate Media Presentation as On-Demand Content.

B.1.2 Use Case 2: Scheduled Service with known duration and Operating at live edge

In this case a service started a few minutes ago and lasts 30 minutes. The duration is known exactly and also all segment URLs are known. The timeshift buffer is short. This may for example be a live service for which the service provider wants to ensure that only a small window is accessible. The content is typically be pre-canned, but offered in a scheduled manner.

B.1.3 Use Case 3: Scheduled Service with known duration and Operating at live edge and time shift buffer

In this case a service started a few minutes ago and lasts 30 minutes. The duration is known exactly and also all segment URLs are known. The timeshift buffer is long. This may for example be a service for which the service provider wants to ensure that the content is made available in a scheduled manner, e.g. no client can access the content earlier than scheduled by the content provider. However, after the live edge is completed, the content is available for 24h. The content is typically pre-canned.

B.1.4 Use Case 4: Scheduled Live Service known duration, but unknown Segment URLs

In this case a live service started a few minutes ago and lasts 30 minutes. The duration is known exactly but the segment URLs are unknown, as for example some advertisement may be added on the fly. Otherwise this service is similar to use case 3.

B.1.5 Use Case 5: 24/7 Live Service

In this case a live service started that may have started a long time ago is made available. Ad breaks and operational updates may be done with a 30sec pre-warning. The duration is unknown and also the segment URLs, the exact set of provided media components (different language tracks, subtitles, etc.) are unknown, as for example some advertisement may be added on the fly. Otherwise this service is similar to use case 3.

B.1.6 Use Case 6: Approximate Media Presentation Duration Known

In this case a live service starts at a specific time. The duration is known approximately and also all segment URLs are known for the approximate duration. Towards the end of the Media Presentation, the Media Presentation duration may be extended or may be finally determined by providing an update of the MPD.

B.2 Baseline Architecture for DASH-based Live Service

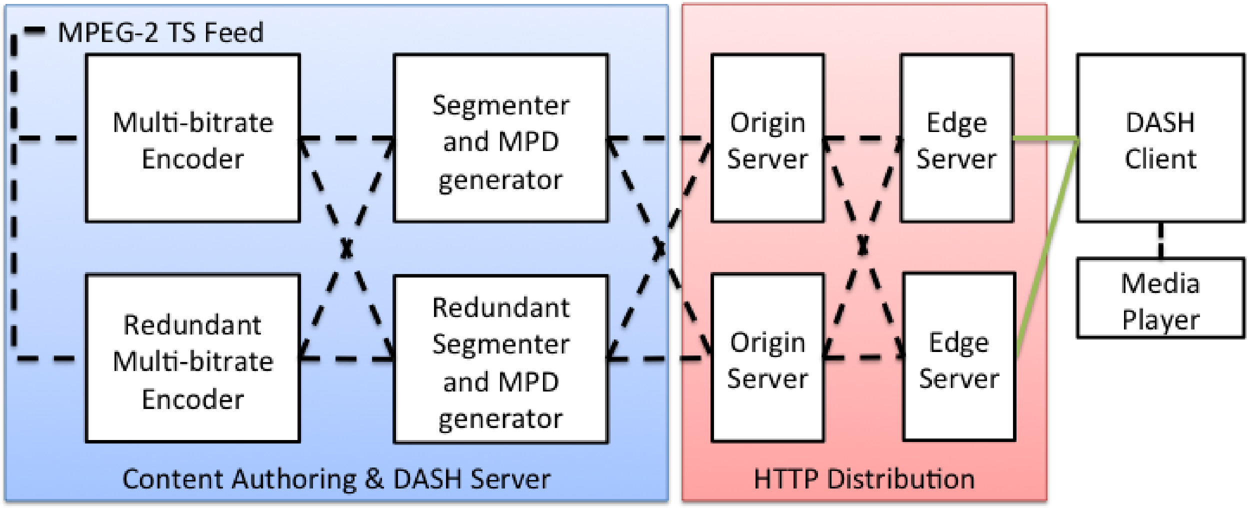

Figure 27 Typical Deployment Scenario for DASH-based live services

The figure depicts a redundant set-up for Live DASH with unicast. Function redundancy is added to mitigate the impact of function failures. The redundant functions are typically connected to multiple downstream functions to mitigate link failure impacts.

An MPEG2-TS stream is used often as feed into the encoder chain. The multi-bitrate encoder produces the required number of Representations for each media component and offers those in one Adaptation Set. In the context of this document is assumed that content is offered in the ISO BMFF live profile with the constraints according to v2 of this document. The encoder typically locks to the system clock from the MPEG2-TS stream. The encoder forwards the content to the segmenter, which produces the actual DASH segments and handles MPD generation and updates. Content Delivery Network (CDN) technologies are typically used to replicate the content to multiple edge servers. Note: the CDN may include additional caching hierarchy layers, which are not depicted here.

Clients fetch the content from edge servers using HTTP (green connection) according to the MPEG-DASH and DASH-IF IOP specification. Different protocols and delivery formats may be used within the CDN to carry the DASH segments from the segmenter to the Edge Server. For instance, the edge server may use HTTP to check with its parent server when a segment is not (yet) in the local cache. Or, segments may be pushed using IP Multicast from the origin server to relevant edge servers. Other realizations are possible, but are outside of the normative scope of this document.

In some deployments, the live service is augmented with ad insertion. In this case, content may not be generated continuously, but may be interrupted by ads. Ads itself may be personalized, targeted or regionalized.

B.3 Distribution over Multicast

This clause describes a baseline architecture for DASH Live Services for broadcast distribution. The intention of the baseline architecture is in particular to identify robustness and failure issue and give guidance on procedures to recover.

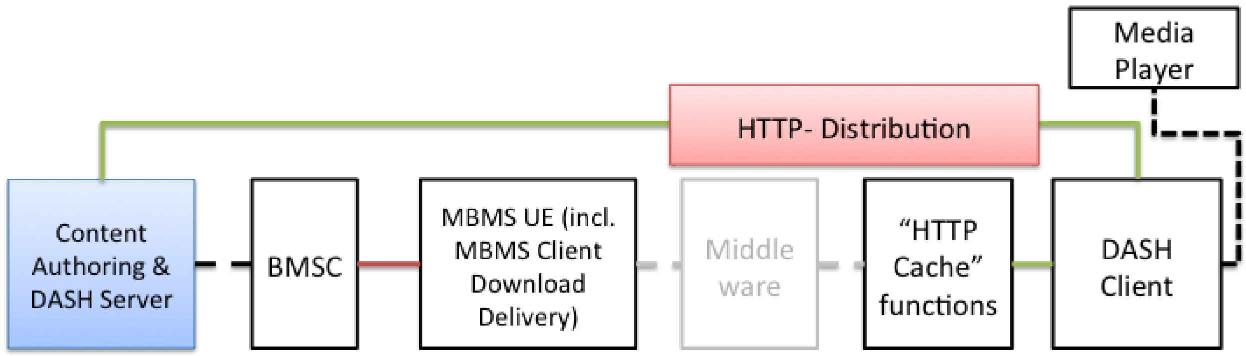

Figure 28 Typical Deployment Scenario for DASH-based live services partially offered through MBMS (uni-directional FLUTE distribution)

The same content authoring and DASH server solution as shown in Figure 1 are considered in this baseline architecture. The DASH Segmenter (cf. Fig .1) provides DASH segments of typically one quality representation into the BM-SC, which sends the segments using MBMS Download (as sequence of files using IETF FLUTE protocol) to the MBMS User Equipment (UE). The MBMS UE includes the needed MBMS download delivery client functions to recover the media segments from the FLUTE reception. The MBMS UE makes the segments through a local HTTP Cache function available to the DASH client. The DASH client uses HTTP (green line) to retrieve the segments from the device local cache.

In case the MBMS reception is not possible for that Video Session, the DASH client can use unicast HTTP to acquire the stream (according to previous clause).

Note, the objective of the client architecture realization here is on using a generic DASH client for unicast and broadcast. More customized implementations are possible.

B.4 Typical Problems in Live Distribution

B.4.1 Introduction

Based on the deployment architectures in Figure 27 and Figure 28 a few typical problems in DASH-based ABR distribution are explained.

B.4.2 Client Server Synchronization Issues

In order to access the DASH segments at the proper time as announced by the segment availability times in the MPD, client and server need to operate in the same time source, in general a globally accurate wall-clock, for example provided by NTP or GPS. There are different reasons why the DASH client and the media generation source may not have identical time source, such as

- DASH client is off because it does not have any protocol access to accurate timing. This may for example be the case for DASH clients that are running in the browser or on top of a general-purpose HTTP stack.
- DASH client clock drifts against the system clock and the DASH client is not synchronizing frequently enough against the time-source.
- The segmenter synchronized against a different time source than DASH client.
- There may be unknown delay on the ingest to the server/cache whether the segment is accessible. This is specifically relevant if MBMS is used as the contribution link resulting in transport delay.

- It may also be that the MPD provides the availability times at the segmenter, but the actual availability should be the one on the origin server.
- There may be a delay from segmenter to the origin server which is known by edge/origin, but there may not be sufficient ways to signal this delay.

B.4.3 Synchronization Loss of Segmenter

The segmenter as depicted in Figure 27 may lose synchronization against the input timeline for reasons such as power-outage, cord cuts, CRC losses in the incoming signals, etc. In this case:

- Loss of synchronization may result that the amount of lost media data cannot be predicted which makes the generation of continuous segments difficult.
- The Segmenter cannot predict and correct the segment timeline based on media presentation timestamps, since the presentation timeline may contain a discontinuity due to the synchronization loss
    - a loss of sync (e.g. CRC failure on the input stream)
    - a power glitch on the source
    - someone pulling a cable
- There are cases where no media segments are available, but the MPD author knows this and just wants to communicate this to the receiver.

B.4.4 Encoder Clock Drift

In certain cases, the MBR encoder is slaved to the incoming MPEG-2 TS, i.e. it reuses the media time stamps also for the ISO BMFF.

- What may occur that the encoder clock drifts between the sender and the receivers (longer term issue) , e.g. due to encoder clock tolerance
    - Example: Encoder produces frame every 39.97ms instead of 40ms
    - Tolerance in MPEG-2 TS: 1 frame every 18 minutes
- This may create issues in particular when an existing stream like for satellite is transcoded and segmented into DASH representations.
- Annex A.8 of ISO 23009-1 handles drift control of the media timeline, but the impact on the segment availability time (i.e. MPD updates) is not considered or suggested.
- In particular when the segment fetching engine of the client is only working with the segment availability timeline (so is not parsing the presentation timeline out of the segments), the segment fetching engine will not fetch the segments with the correct interval, leading to buffer underruns or increased e2e delay.
- There is practical evidence that this is a problem in actual deployments, may result in drifts of minutes over hours.

B.4.5 Segment Unavailability

When a server cannot serve a requested segment it gives an HTTP 404 response. If the segment URL is calculated according to the information given in the MPD, the client can often interpret the 404 response as a possible synchronization issue, i.e. its time is not synchronized to the time offered in the MPD.

In the MBMS case, a 404 response is also likely to be caused by non-reparable transport errors. This is even more likely if it has been possible to fetch segments according to the MPD information earlier. Although the client M/W, which is normally located in the same device as the DASH player, knows what segments have been delivered via broadcast and which ones are missing in a sequence, it cannot indicate this to the DASH client using standard HTTP responses to requests for media segments.

B.4.6 Swapping across Redundant Tools

In case of failures, redundant tools kick in. If the state is not fully maintained across redundant tools, the service may not be perceived continuous by DASH client. Problems that may happen at the encoder, that redundant encoders do not share the same timeline or the timeline is interrupted. Depending on the swap strategy ("hot" or "warm"), the interruptions are more or less obvious to the client. Similar issues may happen if segmenters fail, for example the state for segment numbering is lost.

B.4.7 CDN Issues

Typical CDN operational issues are the following:

- Cache Poisoning – at times segment generation may be erroneous. The encoder can produce a corrupt segment, or the segment can become corrupted during upload to origin. This can happen for example if encoder connectivity fails in mid segment upload, leading to a malformed segment (with the correct name) being sent to edge and caching servers. The CDN then caches this corrupt segment and continues to deliver it to fulfill future requests, leading to widespread client failures.
- Cache inconsistency – with a dual origin scheme, identically named segments can be produced with slight differences in media time, due to clock drift or other encoder issues. These segments are then cached by CDNs and used to respond to client requests. If segments from one encoder are mixed with segments of another, it can lead to discontinuous playback experiences on the clients.

B.4.8 High End-to-end Latency

End-to-end latency (also known as hand-waving latency) is defined as the accumulated delay between an action occurring in front of the camera and that action being visible in a buffered player. It is the sum of

1. Encoder delay in generating a segment.
2. Segment upload time to origin server from the encoder.
3. Edge server segment retrieval time from origin
4. Segment retrieval time by the player from the edge server
5. The distance back from the live point at which the player chooses to start playback.
6. Buffering time on the player before playback commences.

In steps 1 through 4, assuming non-chunked HTTP transfer, the delay is a linear function of the segment duration. Overly conservative player buffering can also introduce unnecessary delay, as can choosing a starting point behind the live point. Generally the further behind live the player chooses to play, the more stable the delivery system is, which leads to antagonistic demands on any production system of low latency and stability.

B.4.9 Buffer Management & Bandwidth Estimation

The main user experience degradations in video streaming are rebuffering events. At the same time, user experience is influenced by the quality of the video (typically determined by the bitrate) as well as at least for certain cases on the end-to-end latency. In order to request the access bitrate, the client does a bandwidth estimation typically based on the history and based on this and the buffer level in the client it decides to maintain or switch Representations.

In order to compensate bandwidth variations, the client buffers some media data prior to play-out. More time buffer results less buffer under runs and less rebuffering, but increases end-to-end latency. In order to maximize the buffer in the client and minimize the end-to-end latency the DASH client would like to request the media segment as close as possible to its actual segment availability start time. However, this may cause issues in the playout as the in case of bitrate variations, the buffer may drain quickly and result in playout starvation and rebuffering.

B.4.10 Start-up Delay and Synchronization Audio/Video

At the start-up and joining, it is relevant that the media playout is initiated, but that the delay at start is reasonable and that the presentation is enabled such that audio and video are presented synchronously. As audio and video Representations typically are offered in different sampling rates, and segments of audio and video are not aligned at segment boundaries. Hence, for proper presentation at startup, it is necessary that the DASH client schedules the presentation at the presentation time aligned to the over media presentation timeline.

B.5 Advanced Use Cases

B.5.1 Introduction

Based on the above issues a few advanced use cases are considered.

B.5.2 Use Case 7: Live Service with undetermined end

In this case a live service started that may have started a long time ago is made available. The MPD update may be done with a 30sec pre-warning. The duration is unknown exactly but the segment URLs are unknown, as for example some advertisement may be added on the fly. Otherwise this service is similar to use case 3.

B.5.3 Use Case 8: 24/7 Live Service with canned advertisement

In this case a live service started that may have started a long time ago is made available. The MPD update may be done with a 30sec pre-warning. The duration is unknown exactly but the segment URLs are unknown, as for example some advertisement may be added on the fly. The advertisement itself is not a dynamic service, but available on a server as a pre-canned advertisement.

B.5.4 Use case 9: 24x7 live broadcast with media time discontinuities

In other use cases, the content provider splices content such as programs and ads with independent media timelines at the content provider.

B.5.5 Use case 10: 24x7 live broadcast with Segment discontinuities

Based on the discussions above, interruptions in encoding, etc., but presentation and media timelines resume after loss of some segments.

Annex C Dolby Vision Streams Within the ISO Base Media File Format

C.1 Introduction

This Annex defines the structures for the storage of Dolby Vision video streams in a file format compliant with the ISO base media file format (ISOBMFF). Example file formats derived from the ISOBMFF include the Digital Entertainment Content Ecosystem (DECE) Common File Format (CFF) and Protected Interoperable File Format (PIFF). Note, that the file format defined here is intended to be potentially compliant with the DECE media specifications as appropriate.

C.2 Dolby Vision Configuration Box and Decoder Configuration Record

The Dolby Vision decoder configuration record provides the configuration information that is required to initialize the Dolby Vision decoder.

C.2.1 Definition

The Dolby Vision Configuration Box contains the following information:

```
Box Type ‘dvcC’

Container
        DolbyVisionHEVCSampleEntry( ‘dvhe’), or
        DolbyVisionHVC1SampleEntry( ‘dvh1’), or

Mandatory Yes

Quantity Exactly One
```

C.2.2 Syntax

The syntaxes of the Dolby Vision Configuration Box and decoder configuration record are described below.

```
align(8) class DOVIDecoderConfigurationRecord
{
  unsigned int (8) dv_version_major;
  unsigned int (8) dv_version_minor;
  unsigned int (7) dv_profile;
  unsigned int (6) dv_level;
  bit (1) dv_metadata_present_flag;
  bit (1) el_present_flag;
  bit (1) bl_present_flag;
  const unsigned int (32)[5] reserved = 0;
}
class DOVIConfigurationBox
```

```
  extends Box('dvcC')
{
  DOVIDecoderConfigurationRecord() DOVIConfig;
}
```

C.2.3 Semantics

The semantics of the Dolby Vision decoder configuration record is described as follows.

dv_version_major - specifies the major version number of the Dolby Vision specification that the stream complies with. A stream compliant with this specification shall have the value 1.

dv_version_minor - specifies the minor version number of the Dolby Vision specification that the stream complies with. A stream compliant with this specification shall have the value 0.

dv_profile – specifies the Dolby Vision profile. Valid values are Profile IDs as defined in Table B.1 of *Signaling Dolby Vision Profiles and Levels,* Annex B.

dv_level – specifies the Dolby Vision level. Valid values are Level IDs as defined in Table B.2 of *Signaling Dolby Vision Profiles and Levels*, Annex B.

dv_metadata_present_flag – if 1 indicates that this track contains the supplemental enhancement information as defined in clause 10.4.2.2.

el_present_flag – if 1 indicates that this track contains the EL HEVC video substream.

bl_present_flag – if 1 indicates that this track contains the BL HEVC video substream.

Note: The settings for these semantic values are specified in Section A.7.1 Constraints on EL Track.

C.3 Dolby Vision Sample Entries

This section describes the Dolby Vision sample entries. It is used to describe tracks that contain substreams that cannot necessarily be decoded by HEVC compliant decoders.

C.3.1 Definition

The Dolby Vision sample entries contain the following information:

Box Type `'dvhe','dvh1'`

Container Sample Description Box (`'stsd'`)

Mandatory Yes

Quantity
One or more sample entries of the same type may be present

C.3.2 Syntax

The syntax for the Dolby Vision sample entries are described below.

```
class DolbyVisionHEVCSampleEntry() extends
    HEVCSampleEntry('dvhe')
{
    DOVIConfigurationBox() config;
}

class DolbyVisionHVC1SampleEntry() extends
    HEVCSampleEntry('dvh1')
{
    DOVIConfigurationBox() config;
}
```

C.3.3 Semantics

A Dolby Vision HEVC sample entry shall contain a Dolby Vision Configuration Box as defined in C.2.2.

config - specifies the configuration information required to initialize the Dolby Vision decoder for a Dolby Vision EL track encoded in HEVC.

Compressorname in the base class VisualSampleEntry indicates the name of the compressor used, with the value "\013DOVI Coding" being recommended (\013 is 11, the length of the string "DOVI coding" in bytes).

C.6 Dolby Vision Files

The brand `'dby1'` SHOULD be used in the compatible_brands field to indicate that the file is compliant with all Dolby Vision UHD Extension as outlined in this document. The `major_brand` shall be set to the ISO-defined brand,e.g. `'iso6'`.

C.7 Dolby Vision Track In A Single File

A Dolby Vision video stream can be encapsulated in a single file as a dual-track file containing separate BL and EL tracks. Each track has different sample descriptions.

C.7.1 Constraints on EL Track

For the visual sample entry box in an EL track a
DolbyVisionHEVCVisualSampleEntry ('`dvhe`') or
DolbyVisionHVC1VisualSampleEntry ('`dvh1`')
shall be used.
The visual sample entries shall contain an HEVC Configuration Box ('`hvcC`') and a Dolby Vision Configuration Box ('`dvcC`').

The EL track shall meet the following constraints:

- In the handler reference box, the `handler_type` field shall be set to '`vide`'.

- The media information header box shall contain a video media header box.
- The dependency between the Dolby Vision base and enhancement track shall be signaled by the '`tref`' box in the enhancement track.. The reference_type shall be set to '`vdep`'.
- The `dv_profile` field in the Dolby Vision Configuration Box ('`dvcC`') shall be set according the encoded Dolby Vision profile.
- The `dv_level` field in the Dolby Vision Configuration Box ('`dvcC`') shall be set according the encoded Dolby Vision level.
- The `dv_metadata_present_flag` shall be set to 1.
- The `el_present_flag` shall be set to 0 or 1.
- The `bl_present_flag` shall be set to 0.

The following table shows the box hierarchy of the EL track.

Note: This is not an exhaustive list of boxes.

Table 30 Sample table box hierarchy for the EL track of a dual-track Dolby Vision file

Nesting Level				Reference
4	5	6	7	
stbl				ISO/IEC 14496-12
	stsd			
		dvhe, or dvh1		Section A.3
			hvcC	
			dvcC	Section 3.1
	stts			ISO/IEC 14496-12
	stsc			
	stsz			
	stz2			
	stco			
	co64			

C.7.2 Constraints on the ISO base media file format boxes

C.7.2.1 Constraints on Movie Fragments

For a dual-track file, the movie fragments carrying the BL and EL shall meet the following constraints:

- The adjacent movie fragments ('`moof`' and '`mdat`') for the base and enhancement track shall be interleaved with BL followed by EL. BL and EL samples shall be placed in separate Movie Fragments and that each BL Movie Fragment shall be immediately followed by an EL movie fragment containing the same number of samples with identical composition timestamps.

C.7.2.2 Constraints on Track Fragment Random Access Box

The track fragment random access box ('`tfra`') for the base and enhancement track shall conform to the ISO/IEC 14496-12 (section 8.8.10) and meet the following additional constraint:

- The value of the `time` field in the track fragment random access box indicates the presentation time of a random accessible sample. This `time` value shall be identical for every corresponding random accessible sample in the base and enhancement track.

Annex D Signaling Dolby Vision Profiles and Levels

This Annex defines the detailed list of Dolby Vision profile/levels and how to represent them in a string format. This string can be used for identifying Dolby Vision device capabilities and identifying the type of the Dolby Vision streams presented to device through various delivery mechanisms such as HTML 5.0 and MPEG-DASH.

D.1 Dolby Vision Profiles and levels

The Dolby Vision codec provides a rich feature set to support various ecosystems such as Over the Top streaming, Broadcast television, Blu-Ray discs, and OTT streaming. The codec also supports many different device implementation types such as GPU accelerated software implementation, full-fledged hardware implementation, and hardware plus software combination. One of the Dolby Vision codec features allows choosing the type of backward compatibility such as non-backward compatible or backward compatible with SDR. A Dolby Vision capable device may not have all the features or options implemented, hence it is critical the device advertises the capabilities and content server provides accurate Dolby vision stream type information.

D.1.1 Dolby Vision Profiles

Following are the currently supported Dolby Vision profiles:

Table D.1: Dolby Vision Profiles

Profile ID	Profile Name	BL Codec	EL Codec	BL:EL	BL Backward Compatibility*	BL/EL Full Alignment**	BL Codec Profile	EL Codec Profile
2	dvhe.der	HEVC8	HEVC8	1:1/4	SDR	No	H.265 Main	H.265 Main
3	dvhe.den	HEVC8	HEVC8	1:1	None	No	H.265 Main	H.265 Main
4	dvhe.dtr	HEVC10	HEVC10	1:1/4	SDR	No	H.265 Main10	H.265 Main10
5	dvhe.stn	HEVC10	N/A	N/A	None	N/A	H.265 Main10	N/A
6	dvhe.dth	HEVC10	HEVC10	1:1/4	HDR10	No	H.265 Main10	H.265 Main10
7	dvhe.dtb	HEVC10	HEVC10	1:1/4 for UHD	Blu-ray HDR	No	H.265 Main10	H.265 Main10
				1:1 for FHD				

Legend:
BL:EL = ratio of Base Layer resolution to Enhancement Layer resolution (when applicable)
BL/EL Full alignment = The Enhancement Layer (EL) GOP and Sub-GOP structures are fully aligned with Base Layer (BL), i.e. the BL/EL IDRs are aligned, BL/EL frames are fully aligned in decode order such that skipping or seeking is possible anywhere in the stream not only limited to IDR. BL AU and EL AU belonging to the same picture shall have the same POC (picture order count)

Encoder Recommendations

* Dolby Vision Encoders should only use baseline profile composer for profiles which are non-backward compatible, i.e. the **BL Backward Compatibility = None**.

** Encoders producing Dolby Vision dual layer streams should generate BL/EL with full GOP/Sub-GOP structure alignment for all the profiles listed in **Table 4.**

D.1.1.1 Dolby Vision Profile String format

The following is the profile string naming convention:

`dv[BL codec type].[number of layers][bit depth][backward compatibility][EL codec type][EL codec bit depth]`

Attribute	Syntax
dv	`dv` = Dolby Vision
BL codec type	`he` = HEVC
Number of layers	`s` = single layer `d` = dual layer without enforcement of BL/EL GOP structure and POC alignment p = dual layer with enforcement of BL/EL GOP structure and POC alignment
Bit depth	`e` = 8 `t` = 10
Backward compatibility	`n` = non-backward compatible `r` = SDR backward compatible (rec.709, 100 nits) `h` = HDR10 backward compatible `b` = Blu-ray backward compatible (Ultra HD Blu-ray™ High Dynamic Range)
EL codec Type ***(see Note 1 below)***	`a` = AVC `h` = HEVC
EL codec bit depth ***(See Note 1 below)***	`e` = 8 `t` = 10

Notes:

1. `[EL codec type]` and `[EL codec bit depth]` shall only be present if the EL codec type is different from the BL codec.
2. Interlaced: There is no support for interlaced video at this time.
3. Codecs other than HEVC or AVC may be supported in future.

D.1.2 Dolby Vision Levels

The Dolby Vision level indicates the maximum frame rate and resolution supported by the device for a given profile. Typically there is a limit on the maximum number of pixels the device can process per second in a given profile; the level indicates the maximum pixels and the maximum bitrate supported in that profile. Since maximum pixels per second is a constant for given level, the resolution can be reduced to get higher frame rate and vice versa. Following are the possible levels:

Table B.2: Dolby Vision Levels

Level ID	Level Name	Example Max Resolution x FPS	Max Bit Rates (BL and EL combined)	
			main tier (Mbps)	high tier (Mbps)
1	hd24	1280x720x24	20	50
2	hd30	1280x720x30	20	50
3	fhd24	1920x1080x24	20	70
4	fhd30	1920x1080x30	20	70
5	fhd60	1920x1080x60	20	70
6	uhd24	3840x2160x24	25	130
7	uhd30	3840x2160x30	25	130
8	uhd48	3840x2160x48	40	130
9	uhd60	3840x2160x60	40	130

B.1.2.1 Dolby Vision Level String Format

The following is the level string naming convention

```
[resolution][FPS][high tier]
```

Attribute	Syntax
Resolution	`hd` = 720 `fhd` = 1080 `uhd` = 2160
FPS	Frames per second (e.g. `24, 30, 60`)
High Tier	Whether or not higher frame rates are supported. If yes, “`h`” will be appended

B.1.3 Dolby Vision Codec Profile and Level String

The profile and level string is recommended to be joined in the following manner:

Format:

```
[Profile String].[Level String]
```

Examples

- `dvav.per.fhd30`
  (dual layer avc 8 bit with enforcement of BL/EL GOP Structure and POC alignment, rec709 backwards compatible, 1920x1080@30fps)

- `dvhe.stn.uhd30`
  (single layer hevc 10 bit non-backwards compatible, 3840x2160@30fps)

B.1.3.1 Device Capabilities

The device capabilities can be expressed in many ways depending on the protocol used by the streaming service or VOD service. The device could maintain a list of supported capabilities in an array:

```
String capabilities [] = {"dvhe.dtr.uhd24", "dvhe.stn.uhd30"}
```

After receiving the manifest the Player could iterate over the stream types and check whether a stream type is supported by searching the capabilities[].

User Agent String

When using HTTP, the device could send the capabilities via the user agent string in HTTP request in following manner:

```
Opera/9.80 (Linux armv71) Presto/2.12.407 Version/12.51 Model-
UHD+dvhe.dtr.uhd24+dvhe.stn.uhd30/1.0.0 (Manufacturer name, Model)
```

A server program can search for "`+dv`" to determine whether Dolby Vision is supported and further identify the profiles and level supported by parsing the characters following the `+dv`. Multiple profiles/level pairs can be listed with '`+`' beginning each profile/level pair.

Annex E Display Management Message

E.1 Introduction

A display management (DM) message contains metadata in order to provide dynamic information about the colour volume of the video signal. This metadata can be employed by the display to adapt the delivered HDR imagery to the capability of the display device. The information conveyed in this message is intended to be adequate for purposes corresponding to the use of Society of Motion Picture and Television Engineers ST 2094-1 and ST 2094-10.

The syntax and semantics for DM_data() are defined in clause C.2.

E.2 Syntax and Semantics

Table C.1: DM_data()

DM_data () {	Descriptor
app_identifier	ue(v)
app_version	ue(v)
metadata_refresh_flag	u(1)
if(metadata_refresh_flag) {	
num_ext_blocks	ue(v)
if(num_ext_blocks) {	
while(!byte_aligned())	
dm_alignment_zero_bit	f(1)
for(i = 0; i < num_ext_blocks; i ++) {	
ext_dm_data_block(i)	
}	
}	
}	
while(!byte_aligned())	
dm_alignment_zero_bit	f(1)
}	

Table C.2: ext_dm_data_block()

ext_dm_data_block() {	Descriptor
ext_block_length	ue(v)
ext_block_level	u(8)

ext_dm_data_block_payload(ext_block_length, ext_block_level)	
}	

Table C.3: ext_dm_data_block_payload()

ext_dm_data_block_payload(ext_block_length, ext_block_level) {	Descriptor
ext_block_len_bits = 8 * ext_block_length	
ext_block_use_bits = 0	
if(ext_block_level == 1) {	
min_PQ	u(12)
max_PQ	u(12)
avg_PQ	u(12)
ext_block_use_bits += 36	
}	
if(ext_block_level == 2) {	
target_max_PQ	u(12)
trim_slope	u(12)
trim_offset	u(12)
trim_power	u(12)
trim_chroma_weight	u(12)
trim_saturation_gain	u(12)
ms_weight	i(13)
ext_block_use_bits += 85	
}	
if(ext_block_level == 5) {	
active_area_left_offset	u(13)
active_area_right_offset	u(13)
active_area_top_offset	u(13)
active_area_bottom_offset	u(13)
ext_block_use_bits += 52	
}	
while(ext_block_use_bits++ < ext_block_len_bits)	
ext_dm_alignment_zero_bit	f(1)
}	

This clause defines the semantics for DM_data().

For the purposes of the present clause, the following mathematical functions apply:

$$\text{Abs}(x) = \begin{cases} x & ; \quad x \geq 0 \\ -x & ; \quad x < 0 \end{cases}$$

Floor(x) is the largest integer less than or equal to x.

$$\text{Sign}(x) = \begin{cases} 1 & ; \quad x > 0 \\ 0 & ; \quad x = 0 \\ -1 & ; \quad x < 0 \end{cases}$$

$$\text{Clip3}(x, y, z) = \begin{cases} x & ; \quad z < x \\ y & ; \quad z > y \\ z & ; \quad otherwise \end{cases}$$

Round(x) = Sign(x)*Floor(Abs(x)+0.5)

/ = Integer division with truncation of the result toward zero. For example, 7/4 and −7/−4 are truncated to 1 and −7/4 and 7/−4 are truncated to −1.

app_identifier identifies an application in the ST 2094 suite.

app_version specifies the application version in the application in the ST 2094 suite.

metadata_refresh_flag when set equal to 1 cancels the persistence of any previous extended display mapping metadata in output order and indicates that extended display mapping metadata follows. The extended display mapping metadata persists from the coded picture to which the SEI message containing DM_data() is associated (inclusive) to the coded picture to which the next SEI message containing DM_data() and with metadata_refresh_flag set equal to 1 in output order is associated (exclusive) or (otherwise) to the last picture in the coded video seqeunce (inclusive). When set equal to 0 this flag indicates that the extended display mapping metadata does not follow.

num_ext_blocks specifies the number of extended display mapping metadata blocks. The value shall be in the range of 1 to 254, inclusive.

dm_alignment_zero_bit shall be equal to 0.

ext_block_length[i] is used to derive the size of the i-th extended display mapping metadata block payload in bytes. The value shall be in the range of 0 to 1023, inclusive.

ext_block_level[i] specifies the level of payload contained in the i-th extended display mapping metadata block. The value shall be in the range of 0 to 255, inclusive. The corresponding extended display mapping metadata block types are defined in Table E.1.4. Values of ext_block_level[i] that are ATSC reserved shall not be present in the bitstreams conforming to this version of ATSC specification. Blocks using ATSC reserved values shall be ignored.

When the value of ext_block_level[i] is set equal to 1, the value of ext_block_length[i] shall be set equal to 5.

When the value of ext_block_level[i] is set equal to 2, the value of ext_block_length[i] shall be set equal to 11.

When the value of ext_block_level[i] is set equal to 5, the value of ext_block_length[i] shall be set equal to 7.

Table C.8: Definition of extended display mapping metadata block type

ext_block_level	extended metadata block type
0	Reserved
1	Level 1 Metadata – Content Range
2	Level 2 Metadata – Trim Pass
3	Reserved
4	Reserved
5	Level 5 Metadata – Active Area
6…255	Reserved

When an extended display mapping metadata block with ext_block_level equal to 5 is present, the following constraints shall apply:

- An extended display mapping metadata block with ext_block_level equal to 5 shall be preceded by at least one extended display mapping metadata block with ext_block_level equal to 1 or 2.
- Between any two extended display mapping metadata blocks with ext_block_level equal to 5, there shall be at least one extended display mapping metadata block with ext_block_level equal to 1 or 2.
- No extended display mapping metadata block with ext_block_level equal to 1 or 2 shall be present after the last extended display mapping metadata block with ext_block_level equal to 5
- The metadata of an extended display mapping metadata block with ext_block_level equal to 1 or 2 shall be applied to the active area specified by the first extended display mapping metadata block with ext_block_level equal to 5 following this block.

When the active area defined by the current extended display mapping metadata block with ext_block_level equal to 5 overlaps with the active area defined by preceding extended display mapping metadata blocks with ext_block_level equal to 5, all metadata of the extended display mapping metadata blocks with ext_block_level equal to 1 or 2 associated with the current extended display mapping metadata block with ext_block_level equal to 5 shall be applied to the pixel values of the overlapping area.

min_PQ specifies the minimum luminance value of the current picture in 12-bit PQ encoding. The value shall be in the range of 0 to 4095, inclusive. Note that the 12-bit min_PQ value with full range is calculated as follows:

$$\text{min\_PQ} = \text{Clip3}(0, 4095, \text{Round}(\text{Min} * 4095))$$

where Min is MinimumPqencodedMaxrgb as defined in clause 6.1.3 of SMPTE ST 2094-10.

max_PQ specifies the maximum luminance value of current picture in 12-bit PQ encoding. The value shall be in the range of 0 to 4095, inclusive. Note that the 12-bit max_PQ value with full range is calculated as follows:

$$\text{max\_PQ} = \text{Clip3}(0, 4095, \text{Round}(\text{Max} * 4095))$$

where Max is MaximumPqencodedMaxrgb as defined in clause 6.1.5 of SMPTE ST 2094-10.

avg_PQ specifies the midpoint luminance value of current picture in 12-bit PQ encoding. The value shall be in the range of 0 to 4095, inclusive. Note that the 12-bit avg_PQ value with full range is calculated as follows:

$$\text{avg\_PQ} = \text{Clip3}(0, 4095, \text{Round}(\text{Avg} * 4095))$$

where Avg is AveragePqencodedMaxrgb as defined in section 6.1.4 of SMPTE ST 2094-10.

target_max_PQ specifies the maximum luminance value of a target display in 12-bit PQ encoding. The value shall be in the range of 0 to 4095, inclusive. The target_max_PQ is the PQ encoded value of TargetedSystemDisplayMaximumLuminance as defined in clause 10.4 of SMPTE ST 2094-1.

If there is more than one extended display mapping metadata block with ext_block_level equal to 2, those blocks shall have no duplicated target_max_PQ.

trim_slope specifies the slope metadata. The value shall be in the range of 0 to 4095, inclusive. If trim_slope is not present, it shall be inferred to be 2048. Note that the 12-bit slope value is calculated as follows:

$$trim\_slope = \mathrm{Clip3}(0, 4095, \mathrm{Round}((S\text{-}0.5) * 4096))$$

where S is the ToneMappingGain as defined in clause 6.2.3 of SMPTE ST 2094-10.

trim_offset specifies the offset metadata. The value shall be in the range of 0 to 4095, inclusive. If trim_offset is not present, it shall be inferred to be 2048. Note that the 12-bit offset value is calculated as follows:

$$trim\_offset = \mathrm{Clip3}(0, 4095, \mathrm{Round}((O{+}0.5) * 4096))$$

where O is the ToneMappingOffset as defined in clause 6.2.2 of SMPTE ST 2094-10.

trim_power specifies the power metadata. The value shall be in the range of 0 to 4095, inclusive. If trim_power is not present, it shall be inferred to be 2048. Note that the 12-bit power value is calculated as follows:

$$trim\_power = \mathrm{Clip3}(0, 4095, \mathrm{Round}((P\text{-}0.5) * 4096))$$

where P is the ToneMappingGamma as defined in clause 6.2.4 of SMPTE ST 2094-10.

trim_chroma_weight specifies the chroma weight metadata. The value shall be in the range of 0 to 4095, inclusive. If trim_chroma_weight is not present, it shall be inferred to be 2048. Note that the 12-bit chroma weight value is calculated as follows:

$$trim\_chroma\_weight = \mathrm{Clip3}(0, 4095, \mathrm{Round}((CW{+}0.5) * 4096))$$

where CW is the ChromaCompensationWeight as defined in clause 6.3.1 of SMPTE ST 2094-10.

trim_saturation_gain specifies the saturation gain metadata. The value shall be in the range of 0 to 4095, inclusive. If trim_saturation_gain is not present, it shall be inferred to be 2048. Note that the 12-bit saturation gain value is calculated as follows:

$$trim\_saturation\_gain = \mathrm{Clip3}(0, 4095, \mathrm{Round}((SG{+}0.5) * 4096))$$

where SG is the SaturationGain as defined in clause 6.3.2 of SMPTE ST 2094-10.

ms_weight specifies the multiscale weight metadata. The value shall be in the range of -1 to 4095, inclusive. If ms_weight is not present, it shall be inferred to be 2048. Where ms_weight is equal to -1, the bit stream indicates ms_weight is unspecified. The 13-bit multiscale weight value is calculated as follows:

$$ms\_weight = \text{-1 OR } \mathrm{Clip3}(0, 4095, \mathrm{Round}(MS * 4096))$$

where *MS* is the ToneDetailFactor as defined in clause 6.4.2 of SMPTE ST 2094-10.

active_area_left_offset, active_area_right_offset, active_area_top_offset, active_area_bottom_offset specify the active area of current picture, in terms of a rectangular region specified in picture coordinates for active area. The values shall be in the range of 0 to 8191, inclusive. See also UpperLeftCorner and LowerRightCorner definitions in ST 2094-1.

If active_area_left_offset, active_area_right_offset, active_area_top_offset, active_area_bottom_offset are not present, they shall be inferred to be 0.

The coordinates of top left active pixel is derived as follows:

$$X_{top\_left} = \text{active\_area\_left\_offset}$$

$$Y_{top\_left} = \text{active\_area\_top\_offset}$$

The coordinates of top left active pixel are defined as the UpperLeftCorner in clause 9.2 of SMPTE ST.2094-1.

With Xsize is the horizontal resolution of the current picture and Ysize is the vertical resolution of current picture, the coordinates of bottom right active pixel are derived as follows:

$$X_{bottom\_right} = \text{XSize} - 1 - \text{active\_area\_right\_offset}$$

$$Y_{bottom\_right} = \text{YSize} - 1 - \text{active\_area\_bottom\_offset}$$

where X_{bottom_right} greater than X_{top_left} and Y_{bottom_right} greater than Y_{top_left}.

The coordinates of bottom right active pixel are defined as the LowerRightCorner in clause 9.3 of SMPTE ST.2094-1.

ext_dm_alignment_zero_bit shall be equal to 0.

Annex F Composing Metadata Message

F.1 Introduction

A composing metadata (CM) message contains the metadata which is needed to apply the post-processing process as described in the ETSI [ETCCM] specification to recreate the HDR UHDTV pictures.

F.2 Syntax and Semantics

The syntax for CM_data() is shown in table D.1. The number of bits “v” used to represent each of the syntax elements of CM_data(), for which the parsing process is specified by the descriptor u(v), is defined in table D.2.

Table D.1: CM_data()

CM_data() {	**Descriptor**
ccm_profile	u(4)
ccm_level	u(4)
coefficient_log2_denom	ue(v)

BL_bit_depth_minus8	ue(v)
EL_bit_depth_minus8	ue(v)
hdr_bit_depth_minus8	ue(v)
disable_residual_flag	u(1)
for(cmp = 0; cmp < 3; cmp++) {	
num_pivots_minus2[cmp]	ue(v)
for (pivot_idx = 0; pivot_idx < num_pivots_minus2[cmp] + 2; pivot_idx + +) {	
pred_pivot_value[cmp][pivot_idx]	u(v)
} // end of pivot points for BL three components	
} //cmp	
for (cmp = 0; cmp < 3; cmp++) { //mapping parameters	
for (pivot_idx = 0; pivot_idx < num_pivots_minus2[cmp] + 1; pivot_idx++) {	
mapping_idc[cmp][pivot_idx]	ue(v)
if(mapping_idc [cmp][pivot_idx] == MAPPING_POLYNOMIAL) {	
poly_order_minus1[cmp][pivot_idx]	ue(v)
for(i = 0 ; i <= poly_order_minus1[cmp][pivot_idx] + 1; i ++) {	
poly_coef_int[cmp][pivot_idx][i]	se(v)
poly_coef[cmp][pivot_idx][i]	u(v)
}	
else if(mapping_idc [cmp][pivot_idx] == MAPPING_MMR) {	
mmr_order_minus1[cmp][pivot_idx]	u(2)
mmr_constant_int[cmp][pivot_idx]	se(v)
mmr_constant[cmp][pivot_idx]	u(v)
for(i = 1; i <= mmr_order_minus1 + 1; i ++) {	
for (j = 0; j < 7; j++) {	
mmr_coef_int[cmp][pivot_idx][i] [j]	se(v)
mmr_coef[cmp][pivot_idx][i][j]	u(v)
} // the j-th coefficients	
} // the i-th order	
} // MMR coefficients	
} // pivot_idx	
} // cmp	
if (!disable_residual_flag) {	
for (cmp = 0; cmp < 3; cmp++) { //quantization parameters	
nlq_offset[cmp]	u(v)
hdr_in_max_int[cmp]	ue(v)
hdr_in_max[cmp]	u(v)
linear_deadzone_slope_int[cmp]	ue(v)
linear_deadzone_slope[cmp]	u(v)
linear_deadzone_threshold_int[cmp]	ue(v)
linear_deadzone_threshold[cmp]	u(v)
} // cmp	

} // disable_residue_flag	
while(!byte_aligned())	
cm_alignment_zero_bit	f(1)
}	

Table D.2: Specification of number of bits "v" for CM_data() syntax elements with descriptor u(v)

Syntax element	Number of bits "v"
pred_pivot_value	EL_bit_depth_minus8 + 8
poly_coef	coefficient_log2_denom
mmr_constant	coefficient_log2_denom
mmr_coef	coefficient_log2_denom
nlq_offset	EL_bit_depth_minus8 + 8
hdr_in_max	coefficient_log2_denom
linear_deadzone_slope	coefficient_log2_denom
linear_deadzone_threshold	coefficient_log2_denom

The definitions of the header parameter values are contained in [ETCCM], Section 5.3.2, "CM Header Parameter Definitions".

The definitions of the mapping parameter values are contained in [ETCCM], Section 5.3.3, "CM Mapping Parameter Definitions".

Parameter **cm_alignment_zero_bit** shall be equal to 0.

Annex G Sample Dual-layer MPD

Below is an example dual-layer MPD, with dual adaptation sets – both a Base layer and an Enhancement Layer. Items of note are highlighted:

```
<Period>
    <!-- Video -->
    <AdaptationSet subsegmentAlignment="true"
subsegmentStartsWithSAP="1" frameRate="24000/1001">
      <Representation mimeType="video/mp4" codecs="
hvc1.2.100000000.L150.B0" id="base-layer"
          bandwidth="14156144" width="3840" height="2160">
        <BaseURL>BL_dual_track_BC.mp4</BaseURL>
        <SegmentBase indexRange="795-1210">
         <Initialization range="0-794"/>
        </SegmentBase>
      </Representation>
      <Representation mimeType="video/mp4" codecs="dvhe.dtr"
id="enhancement-layer"
```

```
                        dependencyId="base-layer" bandwidth="3466528"
width="1920" height="1080">
        <BaseURL>EL_dual_track_BC.mp4</BaseURL>
        <SegmentBase indexRange="704-1119">
         <Initialization range="0-703"/>
        </SegmentBase>
      </Representation>
    </AdaptationSet>
    <!-- Audio -->
    <AdaptationSet mimeType="audio/mp4" codecs="ec-3" lang="und"
        subsegmentAlignment="true" subsegmentStartsWithSAP="1">
      <Representation id="2" bandwidth="192000">
        <AudioChannelConfiguration

schemeIdUri="tag:dolby.com,2014:dash:audio_channel_configuration:20
11" value="F801"/>
        <BaseURL>audio.mp4</BaseURL>
        <SegmentBase indexRange="652-875">
         <Initialization range="0-651"/>
        </SegmentBase>
      </Representation>
    </AdaptationSet>
  </Period>
</MPD>
```