

# Guidelines for Implementation: DASH-IF Interoperability Points

---

December 24, 2015

DASH Industry Forum

*Version 3.2*





# 1 Scope

---

2 The scope of the DASH-IF InterOperability Points (IOPs) defined in this document is to provide support for  
3 high-quality video distribution for over the top services using H.264/AVC and H.265/HEVC. Both live and  
4 on-demand services are supported. The specified features enable relevant use cases including on-demand,  
5 live services, ad insertion, trick modes, content protection and subtitling. Extensions for multi-channel audio  
6 are defined.

7 Any identified bugs or missing features may be submitted through the DASH-IF issue tracker at [https://gi-](https://github.com/Dash-Industry-Forum/DASH-IF-IOP)  
8 [treports.com/issue/Dash-Industry-Forum/DASH-IF-IOP](https://github.com/Dash-Industry-Forum/DASH-IF-IOP).

---

# 1 Disclaimer

---

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

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

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

---

# 1 Contents

---

2	<b>GUIDELINES FOR IMPLEMENTATION: DASH-IF INTEROPERABILITY POINTS</b> .....	1
3	<b>SCOPE</b> .....	1
4	<b>DISCLAIMER</b> .....	2
5	<b>CONTENTS</b> .....	3
6	<b>LIST OF FIGURES</b> .....	7
7	<b>LIST OF TABLES</b> .....	8
8	<b>ACRONYMS, ABBREVIATIONS AND DEFINITIONS</b> .....	9
9	<b>REFERENCES</b> .....	12
10	<b>1. INTRODUCTION</b> .....	1
11	<b>2. CONTEXT AND CONVENTIONS</b> .....	2
12	2.1. RELATION TO MPEG-DASH AND OTHER DASH SPECIFICATIONS .....	2
13	2.2. COMPATIBILITY AND EXTENSIONS TO EARLIER VERSIONS.....	3
14	2.2.1. <i>Summary of Version 3 Modifications</i> .....	3
15	2.2.2. <i>Backward-Compatibility Considerations</i> .....	4
16	2.3. USE OF KEY WORDS .....	4
17	2.3.1. <i>Background</i> .....	4
18	2.3.2. <i>Key Words</i> .....	4
19	2.3.3. <i>Mapping to DASH-IF Assets</i> .....	5
20	2.4. DEFINITION AND USAGE OF INTEROPERABILITY POINTS.....	5
21	2.4.1. <i>Profile Definition in ISO/IEC 23009-1</i> .....	5
22	2.4.2. <i>Usage of Profiles</i> .....	6
23	2.4.3. <i>Interoperability Points and Extensions</i> .....	7
24	<b>3. DASH-RELATED ASPECTS</b> .....	7
25	3.1. SCOPE .....	7
26	3.2. DASH FORMATS.....	7
27	3.2.1. <i>Introduction</i> .....	7
28	3.2.2. <i>Media Presentation Description constraints for v1 &amp; v2 Clients</i> .....	9
29	3.2.3. <i>Segment format constraints</i> .....	10
30	3.2.4. <i>Presence of Attributes and Elements</i> .....	10
31	3.2.5. <i>MPD Dimension Constraints</i> .....	11
32	3.2.6. <i>Generic Metadata</i> .....	11
33	3.2.7. <i>DASH Timing Model</i> .....	12
34	3.2.8. <i>Bandwidth and Minimum Buffer Time</i> .....	14
35	3.2.9. <i>Trick Mode Support</i> .....	15
36	3.2.10. <i>Adaptation Set Constraints</i> .....	15
37	3.2.11. <i>Media Time Information of Segment</i> .....	16
38	3.2.12. <i>Content Offering with Periods</i> .....	18
39	3.3. CLIENT IMPLEMENTATION REQUIREMENTS AND GUIDELINES .....	19
40	3.3.1. <i>Overview</i> .....	19
41	3.3.2. <i>DASH Client Guidelines</i> .....	19
42	3.3.3. <i>Seamless switching</i> .....	20
43	3.3.4. <i>DASH Client Requirements</i> .....	20
44	3.4. TRANSPORT AND PROTOCOL-RELATED ISSUES.....	20
45	3.4.1. <i>General</i> .....	20

1	3.4.2.	<i>Server Requirements and Guidelines</i> .....	21
2	3.4.3.	<i>Client Requirements and Guidelines</i> .....	21
3	3.4.4.	<i>Transforming Proxies and Other Adaptation Middleboxes</i> .....	21
4	3.5.	SYNCHRONIZATION CONSIDERATIONS .....	22
5	3.6.	CONSIDERATIONS FOR LIVE SERVICES.....	22
6	3.7.	CONSIDERATIONS ON AD INSERTION .....	22
7	3.8.	SWITCHING ACROSS ADAPTATION SETS.....	22
8	<b>4.</b>	<b>LIVE SERVICES</b> .....	<b>23</b>
9	4.1.	INTRODUCTION .....	23
10	4.2.	OVERVIEW DYNAMIC AND LIVE MEDIA PRESENTATIONS .....	24
11	4.3.	DYNAMIC SEGMENT DOWNLOAD.....	26
12	4.3.1.	<i>Background and Assumptions</i> .....	26
13	4.3.2.	<i>Preliminaries</i> .....	26
14	4.3.3.	<i>Service Offering Requirements and Guidelines</i> .....	32
15	4.3.4.	<i>Client Operation, Requirements and Guidelines</i> .....	42
16	4.3.5.	<i>Additional DVB-DASH alignment aspects</i> .....	46
17	4.3.6.	<i>Considerations on live edge</i> .....	47
18	4.4.	SIMPLE LIVE SERVICE OFFERING INCLUDING MPD UPDATES.....	47
19	4.4.1.	<i>Background and Assumptions</i> .....	47
20	4.4.2.	<i>Preliminaries</i> .....	48
21	4.4.3.	<i>Service Offering Requirements and Guidelines</i> .....	49
22	4.4.4.	<i>MPD-based Live Client Operation based on MPD</i> .....	53
23	4.5.	MPD AND SEGMENT-BASED LIVE SERVICE OFFERING.....	54
24	4.5.1.	<i>Preliminaries</i> .....	54
25	4.5.2.	<i>Service Offering Requirements and Guidelines</i> .....	56
26	4.5.3.	<i>Client Requirements and Guidelines</i> .....	59
27	4.6.	PROVISIONING OF LIVE CONTENT IN ON-DEMAND MODE.....	61
28	4.6.1.	<i>Scenario</i> .....	61
29	4.6.2.	<i>Content Offering Requirements and Recommendations</i> .....	61
30	4.6.3.	<i>Client Behavior</i> .....	62
31	4.7.	AVAILABILITY TIME SYNCHRONIZATION BETWEEN CLIENT AND SERVER .....	62
32	4.7.1.	<i>Background</i> .....	62
33	4.7.2.	<i>Service Provider Requirements and Guidelines</i> .....	62
34	4.7.3.	<i>Client Requirements and Guidelines</i> .....	63
35	4.8.	ROBUST OPERATION.....	64
36	4.8.1.	<i>Background</i> .....	64
37	4.8.2.	<i>Tools for Robust Operations</i> .....	64
38	4.8.3.	<i>Synchronization Loss of Segmenter</i> .....	65
39	4.8.4.	<i>Encoder Clock Drift</i> .....	65
40	4.8.5.	<i>Segment Unavailability</i> .....	65
41	4.8.6.	<i>Swapping across Redundant Tools</i> .....	65
42	4.8.7.	<i>Service Provider Requirements and Guidelines</i> .....	66
43	4.8.8.	<i>Client Requirements and Guidelines</i> .....	66
44	4.9.	INTEROPERABILITY ASPECTS .....	66
45	4.9.1.	<i>Introduction</i> .....	66
46	4.9.2.	<i>Simple Live Operation</i> .....	66
47	4.9.3.	<i>Main Live Operation</i> .....	67

1	<b>5. AD INSERTION IN DASH</b> .....	<b>67</b>
2	5.1. INTRODUCTION.....	67
3	5.1.1. <i>General</i> .....	67
4	5.1.2. <i>Definitions</i> .....	68
5	5.1.3. <i>DASH Concepts</i> .....	68
6	5.2. ARCHITECTURES.....	71
7	5.3. SERVER-BASED ARCHITECTURE .....	72
8	5.3.1. <i>Introduction</i> .....	72
9	5.3.2. <i>Mapping to DASH</i> .....	72
10	5.3.3. <i>Workflows</i> .....	75
11	5.3.4. <i>Examples</i> .....	79
12	5.4. APP-BASED ARCHITECTURE .....	81
13	5.4.1. <i>Introduction</i> .....	81
14	5.4.2. <i>Mapping to DASH</i> .....	82
15	5.4.3. <i>Workflows</i> .....	83
16	5.5. EXTENSIONS FOR AD INSERTION .....	84
17	5.5.1. <i>Asset Identifiers</i> .....	84
18	5.5.2. <i>Remote Periods</i> .....	85
19	5.5.3. <i>User-defined events</i> .....	85
20	5.6. INTEROPERABILITY ASPECTS .....	85
21	5.6.1. <i>Server-based Ad insertion</i> .....	85
22	5.6.2. <i>App-based Ad Insertion</i> .....	86
23	<b>6. MEDIA CODING TECHNOLOGIES</b> .....	<b>86</b>
24	6.1. INTRODUCTION.....	86
25	6.2. VIDEO .....	86
26	6.2.1. <i>General</i> .....	86
27	6.2.2. <i>DASH-specific aspects for H.264/AVC video</i> .....	87
28	6.2.3. <i>DASH-specific aspects for H.265/HEVC video</i> .....	87
29	6.2.4. <i>Video Metadata</i> .....	88
30	6.2.5. <i>Adaptation Sets Constraints</i> .....	88
31	6.3. AUDIO.....	90
32	6.3.1. <i>General</i> .....	90
33	6.3.2. <i>DASH-specific aspects for HE-AACv2 audio</i> .....	91
34	6.3.3. <i>Audio Metadata</i> .....	91
35	6.4. AUXILIARY COMPONENTS .....	92
36	6.4.1. <i>Introduction</i> .....	92
37	6.4.2. <i>Subtitles and Closed Captioning</i> .....	92
38	6.4.3. <i>CEA-608/708 in SEI messages</i> .....	92
39	6.4.4. <i>Timed Text (IMSC1)</i> .....	95
40	6.4.5. <i>Annotation of Subtitles</i> .....	96
41	<b>7. CONTENT PROTECTION AND SECURITY</b> .....	<b>96</b>
42	7.1. INTRODUCTION.....	96
43	7.2. HTTPS AND DASH .....	96
44	7.3. BASE TECHNOLOGIES SUMMARY.....	98
45	7.4. ISO BMFF SUPPORT FOR COMMON ENCRYPTION AND DRM .....	99
46	7.4.1. <i>Box Hierarchy</i> .....	99
47	7.4.2. <i>ISO BMFF Structure Overview</i> .....	101

1	7.5.	PERIODIC RE-AUTHORIZATION .....	102
2	7.5.1.	<i>Introduction</i> .....	102
3	7.5.2.	<i>Use Cases and Requirements</i> .....	102
4	7.5.3.	<i>Implementation Options</i> .....	102
5	7.6.	MPD SUPPORT FOR ENCRYPTION AND DRM SIGNALING .....	104
6	7.6.1.	<i>Introduction</i> .....	104
7	7.6.2.	<i>Use of the Content Protection Descriptor</i> .....	104
8	7.7.	ADDITIONAL CONTENT PROTECTION CONSTRAINTS .....	105
9	7.7.1.	<i>ISO BMFF Content Protection Constraints</i> .....	105
10	7.7.2.	<i>MPD Content Protections Constraints</i> .....	106
11	7.7.3.	<i>Other Content Protections Constraints</i> .....	107
12	7.7.4.	<i>Additional Constraints for Periodic Re-Authorization</i> .....	107
13	7.7.5.	<i>Encryption of Different Representations</i> .....	107
14	7.7.6.	<i>Encryption of Multiple Periods</i> .....	108
15	7.7.7.	<i>DRM System Identification</i> .....	108
16	7.7.8.	<i>Protection of Media Presentations that Include SD, HD and UHD Adaptation Sets</i> .....	108
17	7.8.	WORKFLOW OVERVIEW .....	110
18	<b>8.</b>	<b>DASH-IF INTEROPERABILITY POINTS .....</b>	<b>113</b>
19	8.1.	INTRODUCTION .....	113
20	8.2.	DASH-AVC/264 MAIN .....	113
21	8.2.1.	<i>Introduction</i> .....	113
22	8.2.2.	<i>Definition</i> .....	113
23	8.3.	DASH-AVC/264 HIGH .....	114
24	8.3.1.	<i>Introduction</i> .....	114
25	8.3.2.	<i>Definition</i> .....	114
26	8.4.	DASH-IF IOP SIMPLE .....	114
27	8.4.1.	<i>Introduction</i> .....	114
28	8.4.2.	<i>Definition</i> .....	114
29	8.5.	DASH-IF IOP MAIN .....	115
30	8.5.1.	<i>Introduction</i> .....	115
31	8.5.2.	<i>Definition</i> .....	115
32	<b>9.</b>	<b>MULTI-CHANNEL AUDIO EXTENSION .....</b>	<b>116</b>
33	9.1.	SCOPE .....	116
34	9.2.	TECHNOLOGIES .....	116
35	9.2.1.	<i>Dolby Multichannel Technologies</i> .....	116
36	9.2.2.	<i>DTS-HD</i> .....	116
37	9.2.3.	<i>MPEG Surround</i> .....	117
38	9.2.4.	<i>MPEG-4 High Efficiency AAC Profile v2, level 6</i> .....	118
39	9.2.5.	<i>MPEG-H 3D Audio</i> .....	118
40	9.3.	CLIENT IMPLEMENTATION GUIDELINES .....	119
41	9.4.	EXTENSIONS .....	119
42	9.4.1.	<i>General</i> .....	119
43	9.4.2.	<i>Dolby Extensions</i> .....	120
44	9.4.3.	<i>DTS-HD Interoperability Points</i> .....	120
45	9.4.4.	<i>MPEG Surround Interoperability Points</i> .....	121
46	9.4.5.	<i>MPEG HE-AAC Multichannel Interoperability Points</i> .....	122
47	9.4.6.	<i>MPEG-H 3D Audio Interoperability Points</i> .....	123



1	<b>ANNEX A</b>	<b>EXAMPLES FOR PROFILE SIGNALLING.....</b>	<b>1</b>
2	<b>ANNEX B</b>	<b>LIVE SERVICES - USE CASES AND ARCHITECTURE .....</b>	<b>2</b>
3	B.1	BASELINE USE CASES.....	2
4	B.1.1	<i>Use Case 1: Live Content Offered as On-Demand .....</i>	<i>2</i>
5	B.1.2	<i>Use Case 2: Scheduled Service with known duration and Operating at live edge .....</i>	<i>2</i>
6	B.1.3	<i>Use Case 3: Scheduled Service with known duration and Operating at live edge and time</i>	
7		<i>shift buffer.....</i>	<i>2</i>
8	B.1.4	<i>Use Case 4: Scheduled Live Service known duration, but unknown Segment URLs .....</i>	<i>2</i>
9	B.1.5	<i>Use Case 5: 24/7 Live Service .....</i>	<i>2</i>
10	B.1.6	<i>Use Case 6: Approximate Media Presentation Duration Known.....</i>	<i>2</i>
11	B.2	BASELINE ARCHITECTURE FOR DASH-BASED LIVE SERVICE .....	3
12	B.3	DISTRIBUTION OVER MULTICAST .....	3
13	B.4	TYPICAL PROBLEMS IN LIVE DISTRIBUTION.....	4
14	B.4.1	<i>Introduction.....</i>	<i>4</i>
15	B.4.2	<i>Client Server Synchronization Issues .....</i>	<i>4</i>
16	B.4.3	<i>Synchronization Loss of Segmenter.....</i>	<i>5</i>
17	B.4.4	<i>Encoder Clock Drift.....</i>	<i>5</i>
18	B.4.5	<i>Segment Unavailability .....</i>	<i>5</i>
19	B.4.6	<i>Swapping across Redundant Tools.....</i>	<i>6</i>
20	B.4.7	<i>CDN Issues .....</i>	<i>6</i>
21	B.4.8	<i>High End-to-end Latency .....</i>	<i>6</i>
22	B.4.9	<i>Buffer Management &amp; Bandwidth Estimation.....</i>	<i>6</i>
23	B.4.10	<i>Start-up Delay and Synchronization Audio/Video.....</i>	<i>7</i>
24	B.5	ADVANCED USE CASES.....	7
25	B.5.1	<i>Introduction.....</i>	<i>7</i>
26	B.5.2	<i>Use Case 7: Live Service with undetermined end .....</i>	<i>7</i>
27	B.5.3	<i>Use Case 8: 24/7 Live Service with canned advertisement .....</i>	<i>7</i>
28	B.5.4	<i>Use case 9: 24x7 live broadcast with media time discontinuities .....</i>	<i>7</i>
29	B.5.5	<i>Use case 10: 24x7 live broadcast with Segment discontinuities .....</i>	<i>7</i>
30			

## 31 List of Figures

32	Figure 1 Overview Timing Model.....	13
33	Figure 2 DASH aspects of a DASH-AVC/264 client compared to a client supporting the union of	
34	DASH ISO BMFF live and on-demand profile.....	19
35	Figure 3 Different Client Models.....	25
36	Figure 4 Segment Availability on the Server for different time NOW (blue = valid but not yet	
37	available segment, green = available Segment, red = unavailable Segment) .....	37
38	Figure 5 Simple Client Model .....	42
39	Figure 6 Advanced Client Model .....	59
40	Figure 7: XLink resolution .....	69
41	Figure 8: Server-based architecture .....	72
42	Figure 9 Using an Asset Identifier .....	74
43	Figure 11: Live Workflow.....	75

---

1	Figure 12: Ad Decision.....	78
2	Figure 13: Example of MPD for "Top Gun" movie .....	81
3	Figure 14: App-based architecture .....	81
4	Figure 15 Inband carriage of SCTE 35 cue message .....	82
5	Figure 16: In-MPD carriage of SCTE 35 cue message.....	83
6	Figure 17: Linear workflow for app-driven architecture .....	83
7	Figure 18: Visualization of box structure for single key content .....	100
8	Figure 19: Visualization of box structure with key rotation .....	100
9	Figure 20: PSSH with version numbers and KIDs. ....	103
10	Figure 21 Logical Roles that Exchange DRM Information and Media.....	110
11	Figure 22 Example of Information flow for DRM license retrieval .....	112
12	Figure 23 Typical Deployment Scenario for DASH-based live services.....	3
13	Figure 24 Typical Deployment Scenario for DASH-based live services partially offered through	
14	MBMS (unidirectional FLUTE distribution).....	4

## 15 List of Tables

---

16	Table 1 DASH-IF Interoperability Points.....	1
17	Table 2 DASH-IF Interoperability Point Extensions .....	1
18	Table 3 Main features and differences of simple and main live services .....	23
19	Table 4 -- Information related to Segment Information and Availability Times for a dynamic service	
20	.....	27
21	Table 5 – Basic Service Offering.....	33
22	Table 6 – Basic Service Offering.....	35
23	Table 7 Multi-Period Service Offering.....	37
24	Table 8 – Service Offering with Segment Timeline.....	40
25	Table 9 – Information related to Live Service Offering with MPD-controlled MPD Updates .....	48
26	Table 10 – Basic Service Offering with MPD Updates.....	50
27	Table 11 – Service Offering with Segment Timeline and MUP greater than 0 .....	52
28	Table 12 – Service Offering with MPD and Segment-based Live Services.....	54
29	Table 13 InbandEventStream@value attribute for scheme with a value	
30	"urn:mpeg:dash:event:2012".....	56
31	Table 14 – Basic Service Offering with Inband Events .....	58
32	Table 15 H.264 (AVC) Codecs parameter according to RFC6381 [11].....	87
33	Table 16 Signaling of HEVC IRAP Pictures in the ISOBMFF and in DASH .....	88
34	Table 17 Codecs parameter according to ISO/IEC 14496-15 [10].....	88
35	Table 18 HE-AACv2 Codecs parameter according to RFC6381 [11].....	91
36	Table 19 Subtitle MIME type and codecs parameter according to IANA and W3C registries.....	95
37	Table 20 Boxes relevant for DRM systems .....	101
38	Table 21 Dolby Technologies: Codec Parameters and ISO BMFF encapsulation .....	116
39	Table 22: DTS Codec Parameters and ISO BMFF encapsulation .....	117

---

1	Table 23 Codecs parameter according to RFC6381 [11] and ISO BMFF encapsulation for MPEG	
2	Surround codec.....	117
3	Table 24 Codecs parameter according to RFC6381 [11] and ISO BMFF encapsulation .....	118
4	Table 25 Codecs parameter and ISO BMFF encapsulation.....	119

5

## 6 Acronyms, abbreviations and definitions

---

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

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

10	AAC	Advanced Audio Coding
11	AFD	Active Format Description
12	AST	Availability Start Time
13	AVC	Advanced Video Coding
14	BMFF	Base Media File Format
15	CDN	Content Delivery Network
16	CEA	Consumer Electronics Association
17	CT	Composition Time
18	DRM	Digital Rights Management
19	DT	Decode Time
20	DTV	Digital Television
21	DVB	Digital Video Broadcasting
22	DVS	Digital Video Subcommittee
23	EDL	Encoding Decision List
24	EME	Encrypted Media Extension
25	EPT	Earliest Presentation Time
26	FCC	Federal Communications Commission
27	GOP	Group-of-Pictures
28	HD	High-Definition
29	HDMI	High-Definition Multimedia Interface

---

1	HE-AAC	High Efficiency AAC
2	HEVC	High-Efficiency Video Coding
3	KID	common Key Identifier
4	IAB	International Advertising Bureau
5	IDR	Instantaneous Decoder Refresh
6	IOP	InterOperability Point
7	ISO	International Standards Organization
8	HEVC	High Efficiency Video Coding
9	HTTP	HyperText Transport Protocol
10	MBT	Minimum Buffer Time
11	MHA	MPEG-H 3D Audio
12	MPEG	Moving Pictures Experts Group
13	MUP	Minimum Update Period
14	NAL	Network Abstraction Layer
15	OTT	Over-The-Top
16	PCM	Pulse Code Modulation
17	PPS	Picture Parameter Set
18	PS	Parametric Stereo
19	PT	Presentation Time
20	PTO	Presentation Time Offset
21	PVR	Personal Video Recorder
22	RFC	Request for Comments
23	SAP	Stream Access Point
24	SAET	Segment Availability End Time
25	SAST	Segment Availability Start Time
26	SBR	Spectral Band Replication
27	SCTE	Society of Cable Telecommunications Engineers
28	SD	Standard Definition
29	SEI	Supplemental Enhancement Information

---

1	SMPTE	Society of Motion Picture and Television Engineers
2	SPD	Suggested Presentation Delay
3	SPS	Sequence Parameter Set
4	TSB	Time Shift Buffer depth
5	TT	Timed Text
6	TTML	Timed Text Markup Language
7	UHD	Ultra-High Definitions
8	URL	Universal Resource Location
9	UTC	Universal Time Clock
10	UUID	Universally Unique Identifier
11	VAST	Video Ad Serving Template
12	VPS	Video Parameter Set

---

# 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*.

- [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/Cor.1:2015/Amd.1:2015 Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment formats.
- [5] *ISO/IEC 23009-2:2015/Amd.3 Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 1: Media presentation description and segment formats.*  
Note: ISO/IEC 23009-1:2014/Amd.3:2015 is expected to be published by Q2/2016. The latest document is available in the MPEG output document w15684.
- [6] ISO/IEC 23009-2:2014: Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 2: Conformance and Reference.
- [7] ISO/IEC 23009-3:2014: Information technology -- Dynamic adaptive streaming over HTTP (DASH) -- Part 3: Implementation Guidelines.
- [8] ISO/IEC 14496-12:2015 Information technology -- Coding of audio-visual objects -- Part 12: ISO base media file format.
- [9] 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".
- [10] 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.
- [11] IETF RFC 6381, The 'Codecs' and 'Profiles' Parameters for "Bucket" Media Types, August 2011.
- [12] 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.
- [13] ISO/IEC 14496-14:2003/Amd 1:2010 Information technology -- Coding of audio-visual objects -- Part 14: The MP4 File Format
- [14] 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

- 
- 1 aacPlus general audio codec; General description (3GPP TS 26.401 version 6.1.0 Release  
2 6)"
- 3 [15] ANSI/CEA-708-E: Digital Television (DTV) Closed Captioning, August 2013
- 4 [16] 3GPP TS 26.245: "Transparent end-to-end Packet switched Streaming Service (PSS);  
5 Timed text format"
- 6 [17] W3C Timed Text Markup Language 1 (TTML1) (Second Edition) 24 September 2013.
- 7 [18] SMPTE ST 2052-1:2013 "Timed Text Format (SMPTE-TT)",  
8 <https://www.smpte.org/standards>
- 9 [19] W3C WebVTT - The Web Video Text Tracks,— <http://dev.w3.org/html5/webvtt/>
- 10 [20] ITU-T Recommendation H.265 (04/2015): "Advanced video coding for generic audiovis-  
11 ual services" | ISO/IEC 23008-2:2014: " High Efficiency Coding and Media Delivery in  
12 Heterogeneous Environments – Part 2: High Efficiency Video Coding", downloadable  
13 here: <http://www.itu.int/rec/T-REC-H.265>
- 14 [21] EBU Tech 3350, "EBU-TT, Part 1, Subtitling format definition", July 2012,  
15 <http://tech.ebu.ch/docs/tech/tech3350.pdf?vers=1.0>
- 16 [22] IETF RFC 7230, Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing,  
17 June 2014.
- 18 [23] IETF RFC 7231, Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content, June  
19 2014.
- 20 [24] IETF RFC 7232, Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests, June  
21 2014.
- 22 [25] IETF RFC 7233, Hypertext Transfer Protocol (HTTP/1.1): Range Requests, June 2014.
- 23 [26] IETF RFC 7234, Hypertext Transfer Protocol (HTTP/1.1): Caching, June 2014.
- 24 [27] IETF RFC 7235, Hypertext Transfer Protocol (HTTP/1.1): Authentication, June 2014.
- 25 [28] SMPTE RP 2052-10-2013: Conversion from CEA-608 Data to SMPTE-TT  
26 <https://www.smpte.org/standards>
- 27 [29] SMPTE RP 2052-11-2013: Conversion from CEA 708 to SMPTE-TT  
28 <https://www.smpte.org/standards>
- 29 [30] ISO/IEC 14496-30:2014, "Timed Text and Other Visual Overlays in ISO Base Media  
30 File Format".
- 31 [31] *ISO/IEC 23001-7:2015: "Information technology -- MPEG systems technologies -- Part  
32 7: Common encryption in ISO base media file format files".*
- 33 Note: ISO/IEC 23001-7:2015 is expected to be published shortly. The latest document is  
34 available in the MPEG output document 14645.
- 35 [32] *DASH Industry Forum, "Guidelines for Implementation: DASH-AVC/264 Test Cases and  
36 Vectors".*
- 37 [33] *DASH Industry Forum, "Guidelines for Implementation: DASH-AVC/264 Conformance  
38 Software", <http://dashif.org/conformance.html>.*
- 39 [34] DASH Identifiers Repository, available here: <http://dashif.org/identifiers>
- 40 [35] DTS 9302J81100, "Implementation of DTS Audio in Media Files Based on ISO/IEC  
41 14496", <http://www.dts.com/professionals/resources/resource-center.aspx>
- 42 [36] ETSI TS 102 366 v1.2.1, Digital Audio Compression (AC-3, Enhanced AC-3) Standard  
43 (2008-08)
- 44 [37] MLP (Dolby TrueHD) streams within the ISO Base Media File Format, version 1.0, Sep-  
45 tember 2009.

- 
- 1 [38] ETSI TS 102 114 v1.3.1 (2011-08), "DTS Coherent Acoustics; Core and Extensions with  
2 Additional Profiles"
- 3 [39] ISO/IEC 23003-1:2007 - Information technology -- MPEG audio technologies -- Part 1:  
4 MPEG Surround
- 5 [40] DTS 9302K62400, "Implementation of DTS Audio in Dynamic Adaptive Streaming over  
6 HTTP (DASH)", <http://www.dts.com/professionals/resources/resource-center.aspx>
- 7 [41] IETF RFC5905, "Network Time Protocol Version 4: Protocol and Algorithms Specifica-  
8 tion," June 2010.
- 9 [42] IETF RFC 6265: "HTTP State Management Mechanism", April 2011.
- 10 [43] ETSI TS 103 285 v.1.1.1: "MPEG-DASH Profile for Transport of ISO BMFF Based DVB  
11 Services over IP Based Networks".
- 12 [44] ANSI/SCTE 128-1 2013: "AVC Video Constraints for Cable Television, Part 1 - Cod-  
13 ing", available here: [http://www.scte.org/documents/pdf/Stand-](http://www.scte.org/documents/pdf/Standards/ANSI_SCTE%20128-1%202013.pdf)  
14 [ards/ANSI\\_SCTE%20128-1%202013.pdf](http://www.scte.org/documents/pdf/Standards/ANSI_SCTE%20128-1%202013.pdf)
- 15 [45] IETF RFC 2119, "Key words for use in RFCs to Indicate Requirement Levels", April  
16 1997.
- 17 [46] ISO: "ISO 639.2, Code for the Representation of Names of Languages — Part 2: alpha-3  
18 code," as maintained by the ISO 639/Joint Advisory Committee (ISO 639/JAC),  
19 <http://www.loc.gov/standards/iso639-2/iso639jac.html>; JAC home page:  
20 <http://www.loc.gov/standards/iso639-2/iso639jac.html>; ISO 639.2 standard online:  
21 <http://www.loc.gov/standards/iso639-2/langhome.html>.
- 22 [47] CEA-608-E, Line 21 Data Service, March 2008.
- 23 [48] IETF RFC 5234, "Augmented BNF for Syntax Specifications: ABNF", January 2008.
- 24 [49] SMPTE ST 2086:2014, "Mastering Display Color Volume Metadata Supporting High  
25 Luminance And Wide Color Gamut Images"
- 26 [50] ISO/IEC 23001-8:2013, "Information technology -- MPEG systems technologies -- Part  
27 8: Coding-independent code points", available here: [http://standards.iso.org/ittf/Public-](http://standards.iso.org/ittf/PubliclyAvailableStandards/c062088_ISO_IEC_23001-8_2013.zip)  
28 [lyAvailableStandards/c062088\\_ISO\\_IEC\\_23001-8\\_2013.zip](http://standards.iso.org/ittf/PubliclyAvailableStandards/c062088_ISO_IEC_23001-8_2013.zip)
- 29 [51] IETF RFC 7164, "RTP and Leap Seconds", March 2014.
- 30 [52] DASH Industry Forum, "Guidelines for Implementation: DASH264 Interoperability  
31 Points", <http://dashif.org/w/2013/08/DASH-AVC-264-v2.00-hd-mca.pdf>.
- 32 [53] IAB Video Multiple Ad Playlist (VMAP), available at [http://www.iab.net/me-](http://www.iab.net/media/file/VMAPv1.0.pdf)  
33 [dia/file/VMAPv1.0.pdf](http://www.iab.net/media/file/VMAPv1.0.pdf)
- 34 [54] IAB Video Ad Serving Template (VAST), available at [http://www.iab.net/me-](http://www.iab.net/media/file/VASTv3.0.pdf)  
35 [dia/file/VASTv3.0.pdf](http://www.iab.net/media/file/VASTv3.0.pdf)
- 36 [55] ANSI/SCTE 35 2015, Digital Program Insertion Cueing Message for Cable
- 37 [56] ANSI/SCTE 67 2014, Recommended Practice for SCTE 35 Digital Program Insertion  
38 Cueing Message for Cable
- 39 [57] ANSI/SCTE 214-1, MPEG DASH for IP-Based Cable Services, Part 1: MPD Constraints  
40 and Extensions
- 41 [58] ANSI/SCTE 214-3, MPEG DASH for IP-Based Cable Services, Part 3: DASH/FF Profile
- 42 [59] EIDR ID Format - EIDR: ID Format, v1.2, March 2014, available at [http://eidr.org/doc-](http://eidr.org/documents/EIDR_ID_Format_v1.2.pdf)  
43 [uments/EIDR\\_ID\\_Format\\_v1.2.pdf](http://eidr.org/documents/EIDR_ID_Format_v1.2.pdf)
- 44 [60] Common Metadata, TR-META-CM, ver. 2.0, January 3, 2013, available at  
45 [http://www.movielabs.com/md/md/v2.0/Common\\_Metadata\\_v2.0.pdf](http://www.movielabs.com/md/md/v2.0/Common_Metadata_v2.0.pdf)
- 46 [61] IETF RFC 4648, "The Base16, Base32, and Base64 Data Encodings", October 2006.



- 
- 1 [62] W3C TTML Profiles for Internet Media Subtitles and Captions 1.0 (IMSC1), Editor’s  
2 Draft 03 August 2015, available at: [https://dvcs.w3.org/hg/ttml/raw-file/tip/ttml-ww-pro-](https://dvcs.w3.org/hg/ttml/raw-file/tip/ttml-ww-profiles/ttml-ww-profiles.html)  
3 [files/ttml-ww-profiles.html](https://dvcs.w3.org/hg/ttml/raw-file/tip/ttml-ww-profiles/ttml-ww-profiles.html)  
4 [63] W3C TTML Profile Registry, available at: [https://www.w3.org/wiki/TTML/CodecsReg-](https://www.w3.org/wiki/TTML/CodecsRegistry)  
5 [istry](https://www.w3.org/wiki/TTML/CodecsRegistry)  
6 [64] ETSI TS 103 190-1 v1.2.1, “Digital Audio Compression (AC-4); Part 1: Channel based  
7 coding”.  
8 [65] ISO/IEC 23008-3:2015 - Information technology -- High efficiency coding and media  
9 delivery in heterogeneous environments -- Part 3: 3D audio with Amendment 1:2015 and  
10 Amendment 2:2015  
11 [66] IETF RFC 2818, “HTTP Over TLS”, May 2000.

1

## 2 1. Introduction

3 This document defines DASH-IF's InterOperability Points (IOPs). The document includes IOPs  
4 for only this version of the document. For earlier versions, please refer to version 1 [1] and version  
5 2 [2] of this document. DASH-IF recommends to deprecate the IOPs in previous versions and  
6 deploy using one of the IOPs and extensions in this document.

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

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

12 In this third version of the DASH-IF IOP document, two new DASH-264/AVC IOPs are defined.  
13 Detailed refinements and improvements for DASH-IF live services and for ad insertion were added  
14 in these IOPs. One of these IOP is the superset of the simpler one. Additionally, two corresponding  
15 IOPs are defined to also support HEVC [20]. In both cases, AVC and HEVC, the more advanced  
16 IOP adds additional requirements on the DASH client to support segment parsing to achieve en-  
17 hancement of live services. This structuring separates the Media Profiles from DASH feature pro-  
18 files.

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

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

23  
24

**Table 1 DASH-IF Interoperability Points**

Interoperability Point	Identifier	Ver- sion	Refer- ence
<b>DASH-AVC/264</b>	<a href="http://dashif.org/guidelines/dash264">http://dashif.org/guidelines/dash264</a>	1.0	[2], 6.3
<b>DASH-AVC/264 SD</b>	<a href="http://dashif.org/guidelines/dash264#sd">http://dashif.org/guidelines/dash264#sd</a>	1.0	[2], 7.3
<b>DASH-AVC/264 HD</b>	<a href="http://dashif.org/guidelines/dash264#hd">http://dashif.org/guidelines/dash264#hd</a>	2.0	[2], 8.3
<b>DASH-AVC/264 main</b>	<a href="http://dashif.org/guidelines/dash264main">http://dashif.org/guidelines/dash264main</a>	3.0	8.2
<b>DASH-AVC/264 high</b>	<a href="http://dashif.org/guidelines/dash264high">http://dashif.org/guidelines/dash264high</a>	3.0	8.3
<b>DASH-IF IOP simple</b>	<a href="http://dashif.org/guidelines/dash-if-simple">http://dashif.org/guidelines/dash-if-simple</a>	3.0	8.4
<b>DASH-IF IOP main</b>	<a href="http://dashif.org/guidelines/dash-if-main">http://dashif.org/guidelines/dash-if-main</a>	3.0	8.5

25

26 Note that all extensions defined in version 2 of this document are carried over into version 3 with-  
27 out any modifications. In order to maintain a single document, referencing in Table 2 is restricted  
28 to this document.

29

**Table 2 DASH-IF Interoperability Point Extensions**

Extension	Identifier	Ver- sion	Section
<b>DASH-IF multichannel audio extension with Enhanced AC-3</b>	<a href="http://dashif.org/guidelines/dashif#ec-3">http://dashif.org/guidelines/dashif#ec-3</a>	2.0	9.4.2.3
<b>DASH-IF multichannel extension with Dolby TrueHD</b>	<a href="http://dashif.org/guidelines/dashif#mlpa">http://dashif.org/guidelines/dashif#mlpa</a>	2.0	9.4.2.3

<b>DASH-IF multichannel audio extension with DTS Digital Surround</b>	<a href="http://dashif.org/guidelines/dashif#dtsc">http://dashif.org/guidelines/dashif#dtsc</a>	2.0	9.4.3.3
<b>DASH-IF multichannel audio extension with DTS-HD High Resolution and DTS-HD Master Audio</b>	<a href="http://dashif.org/guidelines/dashif#dtsh">http://dashif.org/guidelines/dashif#dtsh</a>	2.0	9.4.3.3
<b>DASH-IF multichannel audio extension with DTS Express</b>	<a href="http://dashif.org/guidelines/dashif#dtse">http://dashif.org/guidelines/dashif#dtse</a>	2.0	9.4.3.3
<b>DASH-IF multichannel extension with DTS-HD Lossless (no core)</b>	<a href="http://dashif.org/guidelines/dashif#dtsl">http://dashif.org/guidelines/dashif#dtsl</a>	2.0	9.4.3.3
<b>DASH-IF multichannel audio extension with MPEG Surround</b>	<a href="http://dashif.org/guidelines/dashif#mps">http://dashif.org/guidelines/dashif#mps</a>	2.0	9.4.4.3
<b>DASH-IF multichannel audio extension with HE-AACv2 level 4</b>	<a href="http://dashif.org/guidelines/dashif#heaac-mc51">http://dashif.org/guidelines/dashif#heaac-mc51</a>	2.0	9.4.5.3
<b>DASH-IF multichannel audio extension with HE-AACv2 level 6</b>	<a href="http://dashif.org/guidelines/dashif#heaac-mc71">http://dashif.org/guidelines/dashif#heaac-mc71</a>	2.0	9.4.5.3
<b>DASH-IF multichannel extension with AC-4</b>	<a href="http://dashif.org/guidelines/dashif#ac-4">http://dashif.org/guidelines/dashif#ac-4</a>	3.1	9.4.2.3
<b>DASH-IF multichannel audio extension with MPEG-H 3D Audio</b>	<a href="http://dashif.org/guidelines/dashif#mha1">http://dashif.org/guidelines/dashif#mha1</a> <a href="http://dashif.org/guidelines/dashif#mha2">http://dashif.org/guidelines/dashif#mha2</a>	3.2	9.4.6.3

1  
2 Test cases and test vectors for DASH-AVC/264 IOPs are provided in [32]. The conformance and  
3 reference software for DASH-IF IOPs is provided in [33] (based on the MPEG conformance soft-  
4 ware [6]). DASH Identifiers for different categories can be found online [34].

## 5 **2. Context and Conventions**

### 6 **2.1. Relation to MPEG-DASH and other DASH specifications**

7 Dynamic Adaptive Streaming over HTTP (DASH) is initially defined in the first edition of  
8 ISO/IEC 23009-1 which was published in April 2012 and some corrections were done in 2013 [1].  
9 In May 2014, ISO/IEC published the second version of ISO/IEC 23009-1 [5] that includes addi-  
10 tional features and provide additional clarifications. The initial two versions of this document  
11 where based on the first edition of ISO/IEC 23009-1. This version is based on the second edition  
12 of ISO/IEC 23009-1, i.e. ISO/IEC 23009-1:2014 including Cor.1 and Amd.1 [5]. This means that  
13 also for all interoperability points that were initially defined in earlier versions of the document,  
14 also now the second edition serves as the reference. Backward-compatibility across different edi-  
15 tion is handled by MPEG-DASH in ISO/IEC 23009-1 [5].

16 This document was generated in close coordination with DVB-DASH [43]. The tools and features  
17 are aligned to the extent considered reasonable. To support implementers, this document attempts  
18 to highlight any differences and/or further restrictions or extensions when compared to DVB-  
19 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 [5].
- Add an explicit statement in DASH-264/AVC to forbid time code wrap around
- Definition on the usage of key words in section 2.3.
- Add more constraints on the usage of Trick Modes for improved interoperability in section 3.2.9.
- Add more constraints on the Representations in one Adaptation Set in section 3.2.10, especially for the case when the bitstream switching is true.
- Add additional details on the usage of HTTP in section 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 section 6.4.3.
- Detailed description of simple and main live operation, with the latter including segment parsing in section 4.
- Detailed description of server-based and app-based ad insertion in section 5
- General editorial updates and clarifications
- Updates and clarification to section 7 on DRM and common encryption.
- Update to references
- Relaxation of the audio encoding requirements in section 6.3.2.
- Add clarification on the usage of the minimum buffer time and bandwidth in section 3.2.8.
- Add an informative section on the timing model of DASH in section 3.2.7.
- Relax the use of the 'lmsg' brand for signaling the last segment in section 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 section is added on how to operate with transforming proxies and other adaptation middle-boxes in section 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 16 based on feedback from MPEG.
- Clarify relation between SMPTE-TT and IMSC1 in section 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 section 3.2.7.
- Added support for switching across Adaptation Sets, in particular for H.264/AVC and H.265/HEVC switching in section 3.8.

- 
- 1 • Added a clarification on the value of @audioSamplingRate for AAC SBR in section
  - 2 6.3.2.
  - 3 • Added a section on the use of HTTPS with DASH in section 7.2.
  - 4 • Moved the test DRM to the test vector document 7.
  - 5 • Added support for key rotation in section 7.
  - 6 • Add extension for MPEG-H audio in section 9.

### 7 **2.2.2. Backward-Compatibility Considerations**

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

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

- 13 • The backward-compatibility across MPEG editions is handled in the second edition of
- 14 ISO/IEC 23009-1 [5].
- 15 • General clarifications and updates are added
- 16 • Further restrictions on content authoring compared to version 2 are:
  - 17 ○ forbid time code wrap around
  - 18 ○ the usage of DRM, especially the Content Protection element
  - 19 ○ constraints on trick mode usage
  - 20 ○ additional constraints on the usage of HTTP
  - 21 ○ Adaptation Set constraints
- 22 • Relaxations are:
  - 23 ○ Permit usage of additional subtitling format based on CEA-608/708
  - 24 ○ the audio encoding requirements for HE-AACv2
  - 25 ○ permit to not use of the 'lmsg' brand for signaling the last segment
  - 26 ○ the ability to signal bitstream switching set to true
  - 27 ○ the use of remote elements with Xlink

## 29 **2.3. Use of Key Words**

### 30 **2.3.1. Background**

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

### 35 **2.3.2. Key Words**

36 The key word usage is aligned with the definitions in RFC 2119 [45], namely:

- 37 • SHALL: This word means that the definition is an absolute requirement of the specifica-  
38 tion.
- 39 • SHALL NOT: This phrase means that the definition is an absolute prohibition of the  
40 specification.

- 
- 1 • SHOULD: This word means that there may exist valid reasons in particular circumstances  
2 to ignore a particular item, but the full implications must be understood and carefully  
3 weighed before choosing a different course.
  - 4 • SHOULD NOT: This phrase means that there may exist valid reasons in particular cir-  
5 cumstances when the particular behavior is acceptable or even useful, but the full implica-  
6 tions should be understood and the case carefully weighed before implementing any be-  
7 havior described with this label.
  - 8 • MAY: This word means that an item is truly optional. One vendor may choose to include  
9 the item because a particular marketplace requires it or because the vendor feels that it  
10 enhances the product while another vendor may omit the same item.

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

### 12 **2.3.3. Mapping to DASH-IF Assets**

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

- 15 • SHALL: The conformance software provides a conformance check for this and issues an  
16 *error* if the conformance is not fulfilled.
- 17 • SHALL NOT: The conformance software provides a conformance check for this and issues  
18 an *error* if the conformance is not fulfilled.
- 19 • SHOULD: The conformance software provides a conformance check for this and issues a  
20 *warning* if the conformance is not fulfilled.
- 21 • SHOULD NOT: The conformance software provides a conformance check for this and  
22 issues a *warning* if the conformance is not fulfilled.
- 23 • SHOULD and MAY: If present, the feature check of the conformance software documents  
24 a feature of the content.

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

- 27 • SHALL: Test content is necessarily provided with this rule and the reference client imple-  
28 ments the feature.
- 29 • SHALL NOT: The reference client does not implement the feature.
- 30 • SHOULD: Test content is provided with this rule and the reference client implements the  
31 feature unless there is a justification for not implementing this.
- 32 • SHOULD NOT: The reference client does not implement the feature unless there is a jus-  
33 tification for implementing this.
- 34 • MAY: Test content is provided and the reference client implements the feature if there is a  
35 justification this.

## 36 **2.4. Definition and Usage of Interoperability Points**

### 37 **2.4.1. Profile Definition in ISO/IEC 23009-1**

38 MPEG DASH defines formats for MPDs and Segments. In addition MPEG provides the ability to  
39 further restrict the applied formats by the definition of *Profiles* as defined on section 8 of ISO/IEC  
40 23009-1 [5]. Profiles of DASH are defined to enable interoperability and the signaling of the use  
41 of features.

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

---

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

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

- 9 • a collection of content conforming points, i.e. as long as the content conforms to the re-  
10 strictions as specified by the IOP, clients implementing the features can consume the con-  
11 tent.
- 12 • a client capability points that enable content and service providers for flexible service pro-  
13 visioning to clients conforming to these client capabilities.

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

#### 17 **2.4.2. Usage of Profiles**

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

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

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

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

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

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

- 37 1. The MPD of the Media Presentation is conforming to profile ProfA as specified above.
- 38 2. There is at least one Representation in each Period in the profile-specific MPD for ProfA.
- 39 3. The Segments of the Representations of the profile-specific MPD for ProfA conform to  
40 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 [8] 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 [8], 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:



- 
- 1 • Segment formats are based on ISO BMFF with fragmented movie files, i.e. (Sub)Segments  
2 are encoded as movie fragments containing a track fragment as defined in ISO/IEC 14496-  
3 12 [8], plus the following constraints to make each movie fragment independently de-  
4 codable:
- 5 • Default parameters and flags shall be stored in movie fragments ('`tfhd`' or '`trun`'  
6 box) and not track headers ('`trex`' box)
  - 7 • The '`moof`' boxes shall not use external data references, the flag '`default-base-`  
8 `is-moof`' shall also be set (aka movie-fragment relative addressing) and `data-`  
9 `offset` shall be used, i.e. `base-data-offset-present` shall not be used (fol-  
10 lows ISO/IEC 23009-1 [4]).
  - 11 • Alignment with ISO BMFF Live & On-Demand Profiles, i.e. within each Adaptation Set  
12 the following applies
    - 13 • Fragmented movie files are used for encapsulation of media data
    - 14 • (Sub)Segments are aligned to enable seamless switching

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

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

25 Note: If accurate seeking to specific time is required and at the same time a fast  
26 response is required one may use On-Demand profile for VoD or the **SegmentTimeline**  
27 based addressing. Otherwise the offset in segment duration com-  
28 pared to the actual media segment duration may result in a less accurate seek posi-  
29 tion for the download request, resulting in some increased initial start-up.

- 30 • If the **SegmentTimeline** element is used for the signaling of the Segment duration, the  
31 timing in the segment timeline shall be media time accurate and no constraints on segment  
32 duration deviation are added except the maximum segment duration as specified in the  
33 MPD. However, despite the usage of the the **SegmentTimeline**, it is not encouraged to  
34 use varying Segment durations. The **SegmentTimeline** element should only be used  
35 in order to signal occasional shorter Segments (possibly caused by encoder processes) or  
36 to signal gaps in the time line.
- 37 • only non-multiplexed Representations shall be used, i.e. each Representation only contains  
38 a single media component.
- 39 • Addressing schemes are restricted to
  - 40 • templates with number-based addressing
  - 41 • templates with time-based addressing
  - 42 • Subsegments with Segment Index. In this case either the `@indexRange` attribute shall  
43 be present or the **RepresentationIndex** element shall be present. Only a single  
44 `sidx` box shall be present.

---

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

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

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

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

## 16 **3.2.2. Media Presentation Description constraints for v1 & v2 Clients**

### 17 **3.2.2.1. Definition according to ISO/IEC 23009-1**

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

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

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

- 26 • Representations with `@mimeType` attribute `application/xml+ttml` shall not be ig-  
27 nored.

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

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

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

32 — If a Period contains multiple Adaptation Sets with `@contentType="video"` then at least  
33 one Adaptation Set shall contain a Role element `<Role schemeId-`  
34 `dUri="urn:mpeg:dash:role:2011" value="main">` and each Adaptation Set  
35 containing such a **Role** element shall provide perceptually equivalent media streams.

### 36 **3.2.2.2. Simple Restricted Content Offering**

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

- 38 • **MPD**`@type` set to `static` or set to `dynamic`.
- 39 • **MPD**`@profiles` includes `urn:mpeg:dash:profile:isoff-live:2011`

- 
- 1 • One or multiple Periods with each containing one or multiple Adaptation Sets and with
  - 2 each containing one or multiple Representations.
  - 3 • The Representations contain or inherit a **SegmentTemplate** with `$Number$` or
  - 4 `$Time$` Identifier.
  - 5 • `@segmentAlignment` set to true for all Adaptation Sets

6

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

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

### 14 3.2.3. Segment format constraints

15 Representations and Segments referred to by the Representations in the profile-specific MPD for

16 this profile, the following constraints shall be met:

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

18 — In Media Segments, all Segment Index ('`sidx`') and Subsegment Index ('`ssix`') boxes, if

19 present, shall be placed before any Movie Fragment ('`moof`') boxes.

20 Note: DVB DASH [43] permits only one single Segment Index box ('`sidx`') for the entire Segment. As this con-

21 straints is not severe in the content offering, it is strongly recommended to offer content following this constraint.

22 — If the **MPD@type** is equal to `"static"` and the **MPD@profiles** attribute includes

23 `"urn:mpeg:dash:profile:isoff-on-demand:2011"`, then

24 — Each Representation shall have one Segment that complies with the Indexed Self-Initial-

25 izing Media Segment as defined in section 6.3.5.2 in ISO/IEC 23009-1.

26 — Time Codes expressing presentation and decode times shall be linearly increasing with in-

27 creasing Segment number in one Representation. In order to minimize the frequency of time

28 code wrap around 64 bit codes may be used or the timescale of the Representation may be

29 chosen as small as possible. In order to support time code wrap around, a new Period may be

30 added in the MPD added that initiates a new Period in order to express a discontinuity.

### 31 3.2.4. Presence of Attributes and Elements

32 Elements and attributes are expected to be present for certain Adaptation Sets and Representations

33 to enable suitable initial selection and switching.

34 Specifically the following applies:

- 35 • For any Adaptation Sets with `@contentType="video"` the following attributes shall
- 36 be present
  - 37 ○ `@maxWidth` (or `@width` if all Representations have the same width)

- 
- 1       ○ @maxHeight (or @height if all Representations have the same height)
  - 2       ○ @maxFrameRate (or @frameRate if all Representations have the same frame
  - 3           rate)
  - 4       ○ @par

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

- 8       • For any Representation within an Adaptation Set with @contentType="video" the  
9       following attributes shall be present:
  - 10       ○ @width, if not present in **AdaptationSet** element
  - 11       ○ @height, if not present in **AdaptationSet** element
  - 12       ○ @frameRate, if not present in **AdaptationSet** element
  - 13       ○ @sar

14       Note: @width, @height, and @sar attributes should indicate the vertical and horizontal  
15       sample count of encoded and cropped video samples, not the intended display size in pixels.

- 16       • For Adaptation Set or for any Representation within an Adaptation Set with @con-  
17       tentType="video" the attribute @scanType shall either not be present or shall be  
18       set to "progressive".
- 19       • For any Adaptation Sets with value of the @contentType="audio" the following at-  
20       tributes shall be present
  - 21       ○ @lang
- 22       • For any Representation within an Adaptation Set with value of the @con-  
23       tentType="audio" the following elements and attributes shall be present:
  - 24       ○ @audioSamplingRate, if not present in **AdaptationSet** element
  - 25       ○ **AudioChannelConfiguration**, if not present in **AdaptationSet** element

### 26   **3.2.5.   MPD Dimension Constraints**

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

29       Note: DVB DASH [43] adds MPD dimension constraints in section 4.5 of their specification. In order to conform  
30       to this specification, it is recommended to obey these constraints.

### 31   **3.2.6.   Generic Metadata**

32   Generic metadata may be added to MPDs based on DASH. For this purpose, the Essential Property  
33   Descriptor and the Supplemental Property Descriptor as defined in ISO/IEC 23009-1 [4], clause  
34   5.8.4.7 and 5.8.4.8.

35   Metadata identifiers for content properties are provided here: [http://dashif.org/iden-](http://dashif.org/identifiers)  
36   [tifiers](http://dashif.org/identifiers).

---

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

### 3 **3.2.7. DASH Timing Model**

#### 4 **3.2.7.1. General**

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

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

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

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

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

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

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

40 The media timeline origin for tracks in ISO Media files is defined to be zero. Each Representation  
41 is assigned a presentation time offset, either by the value of the attribute **Representa-**

1 **tion**@presentationTimeOffset or by default set to 0. The value of this attribute is de-  
 2 noted as PTO. It is normally the case that for complete files sequenced as Periods this value is 0.  
 3 For “partial” files or live streams, @presentationTimeOffset indicates the media compo-  
 4 sition/presentation time of the samples that synchronize to the start of the Period. @presenta-  
 5 tionTimeOffset for the live stream will usually not be zero because the encoders are usually  
 6 started prior to presentation availability, so the media timestamps on the first available Segments  
 7 will have increased since the encoders started. Encoding timestamps may be set to UTC (as though  
 8 the encoder was turned on 1/1/1970 at midnight). Representations in Periods of live content typi-  
 9 cally have the same @presentationTimeOffset as long as the media is continuously en-  
 10 coded, because UTC time and media time increase at the same rate and maintain the same offset.

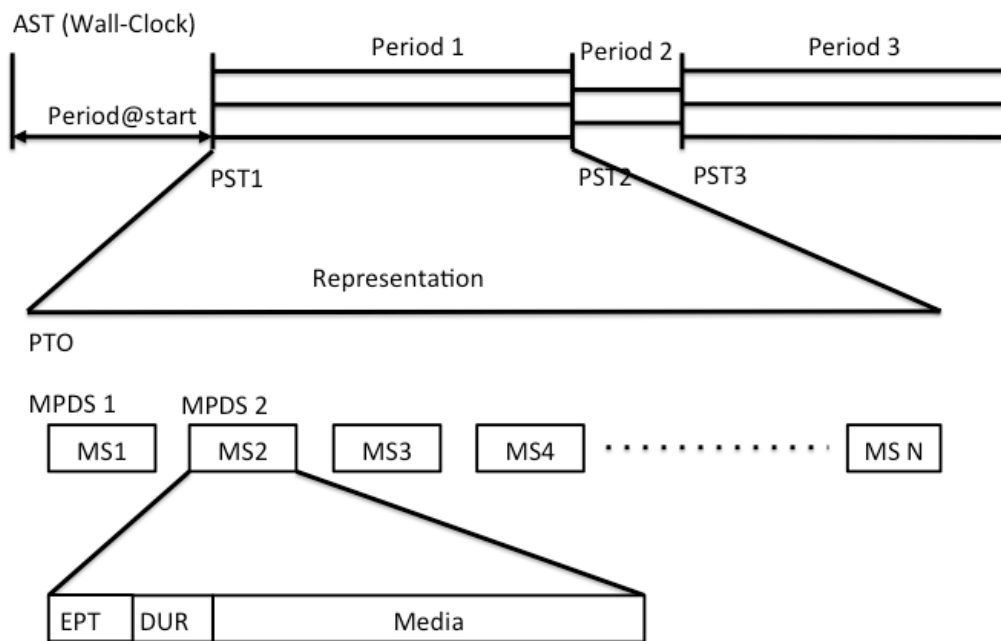


Figure 1 Overview Timing Model

11  
 12  
 13 Within a Representation, each Segment is assigned an MPD start time and MPD duration accord-  
 14 ing to ISO/IEC 23009-1 (more details for dynamic services are provided in section 4). These two  
 15 values can be computed from the MPD and provide approximate times for each segment that are  
 16 in particular useful for random access and seeking.

17 In addition, each segment has an internal sample-accurate presentation time. Therefore, each seg-  
 18 ment has a media internal earliest presentation time *EPT* and sample-accurate presentation dura-  
 19 tion *DUR*.

20 For each media segment in each Representation the MPD start time of the segment should approx-  
 21 imately 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.

22  
 23 Each Period is treated independently. Details on processing at Period boundaries are provided in  
 24 ISO/IEC 23009-1. One example is, that for time code wrap-around a new Period is expected to be  
 25 added that restarts at presentation time 0.

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

---

### 1 3.2.7.2. Static Media Presentations

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

### 4 3.2.7.3. Dynamic Media Presentations

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

### 9 3.2.8. Bandwidth and Minimum Buffer Time

10 The MPD contains a pair of values for a bandwidth and buffering description, namely the Mini-  
11 mum Buffer Time (MBT) expressed by the value of  $\mathbf{MPD@minBufferTime}$  and bandwidth (BW)  
12 expressed by the value of  $\mathbf{Representation@bandwidth}$ . The following holds:

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

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

- 31 • the currently available buffer in the media pipeline, *buffer*
- 32 • the currently estimated download rate, *rate*
- 33 • the value of the attribute  $\mathbf{@minBufferTime}$ , *MBT*
- 34 • the set of values of the  $\mathbf{@bandwidth}$  attribute for each Representation *i*,  $BW[i]$

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

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

40 Note that in this model, some idealizations typically do not hold in practice, such as constant bitrate  
41 channel, progressive download and playout of Segments, no blocking and congestion of other  
42 HTTP requests, etc. Therefore, a DASH client should use these values with care to compensate

---

1 such practical circumstances; especially variations in download speed, latency, jitter, scheduling  
2 of requests of media components, as well as to address other practical circumstances.  
3 One example is if the DASH client operates on Segment granularity. As in this case, not only parts  
4 of the Segment (i.e., MBT) needs to be downloaded, but the entire Segment, and if the MBT is  
5 smaller than the Segment duration, then rather the segment duration needs to be used instead of  
6 the MBT for the required buffer size and the download scheduling, i.e. download a Representation  
7  $i$  for which  $BW[i] \leq rate * buffer / max\_segment\_duration$ .

### 8 **3.2.9. Trick Mode Support**

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

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

- 17 • add an Adaptation Set that that only contains trick modes Representations
- 18 • annotate the Adaptation Set with an **EssentialProperty** descriptor or **Supple-**  
19 **mentalProperty** descriptor with URL "[http://dashif.org/guide-](http://dashif.org/guidelines/trickmode)  
20 [lines/trickmode](http://dashif.org/guidelines/trickmode)" and the @value the value of @id attribute of the Adaptation Set  
21 to which these trick mode Representations belong. The trick mode Representations must  
22 be time-aligned with the Representations in the main Adaptation Set. The value may also  
23 be a white-space separated list of @id values. In this case the trick mode Adaptation Set is  
24 associated to all Adaptation Sets with the values of the @id.
- 25 • signal the playout capabilities with the attribute @maxPlayoutRate for each Represent-  
26 ation in order to indicate the accelerated playout that is enabled by the signaled codec  
27 profile and level.

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

### 31 **3.2.10. Adaptation Set Constraints**

#### 32 **3.2.10.1. Introduction**

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

38 Bitstream Switching Adaptation Set constraints allow a switched sequence of Media Segments to  
39 be decoded without resetting the decoder at switch points because the resulting Segment stream is  
40 a valid track of the source type, so the decoder is not even aware of the switch. In order to signal  
41 that the Representations in an Adaptation Set are offered under these constraints, the attribute **Ad-**



---

1 `aptationSet@bitstreamSwitching` may be set to `true`. In the following general re-  
2 quirements and recommendations are provided for content in an Adaptation Set in section 3.2.10.2  
3 and specific requirements when the bitstream switching is set to true in section 3.2.10.3.

#### 4 **3.2.10.2. General**

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

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

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

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

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

#### 19 **3.2.10.3. Bitstream Switching**

20 A bitstream switching Adaptation Set is optimized for seamless decoding, and live streams that  
21 may change encoding parameters over time. A bitstream switching Adaptation Set may process an  
22 Initialization Segment one time from the highest bandwidth Representation in the Adaptation Set,  
23 and then process Media Segments from any other Representation in the same Adaptation Set with-  
24 out processing another Initialization Segment. The resulting sequence of an Initialization Segment  
25 followed by time sequenced Media Segments results in a valid ISO BMFF file with an elementary  
26 stream similar to a transport stream.

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

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

30 Note: Multiple Adaptation Sets may be included in an MPD that contain different subsets of the available  
31 Representations that are optimized for different decoder and screen limitations. A Representation may be  
32 present in more than one Adaptation set, for example a 720p Representation that is present in a 720p Adap-  
33 tation Set may also be present in a 1080p Adaptation Set. The 720p Representation uses the same Initializa-  
34 tion Segments in each Adaptation Set, but the 1080p Adaptation Set would require decoder and display con-  
35 figuration with the 1080p Initialization Segment.

36 Additional recommendation and constraints may apply for encryption and media coding. For de-  
37 tails, please see below.

#### 38 **3.2.11. Media Time Information of Segment**

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

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

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

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

4 The earliest presentation time in the Period for a Segment can be deduced from the decode time  
 5 taking also into account the composition time offset, edit lists as well as presentation time offsets.  
 6 Specifically the following is the case to determine the earliest presentation time assuming that no  
 7 edit list is present in the Initialization Segment:

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

17  
 18 In addition, if the presentation time needs to be adjusted at the beginning of a period, then the  
 19 @presentationTimeOffset shall be used in order to set the presentation that is mapped to  
 20 the start of the period. Content authoring shall be such that if edit lists are ignored, then the client  
 21 can operate without timing and lip sync issues.

22  
 23 In the following examples, there is a sequence of I, P, and B frames, each with a decoding time  
 24 delta of 10. The segmentation, presentation order and storage of the samples is shown in the table  
 25 below. The samples are stored with the indicated values for their decoding time deltas and com-  
 26 position time offsets (the actual CT and DT are given for reference). The re-ordering occurs be-  
 27 cause the predicted P frames must be decoded before the bi-directionally predicted B frames. The  
 28 value of DT for a sample is always the sum of the deltas of the preceding samples. Note that the  
 29 total of the decoding deltas is the duration of the media in this track.

30  
 31 Example with closed GOP and SAP Type = 1:  
 32

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
----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----

1  
2  
3

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

4  
5  
6

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
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	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

7  
8  
9  
10

For additional details refer to ISO/IEC 14496-12 [8] and ISO/IEC 23009-1 [1].

### 11 3.2.12. Content Offering with Periods

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

14 However, other values for PSTART may be chosen.

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

- to enable splicing of content, for example for ad insertion,
- to remove or add certain Representations in an Adaptation Set,
- to remove or add certain Adaptation Sets,
- for robustness reasons as documented in detail in section 4.8.

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

### 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

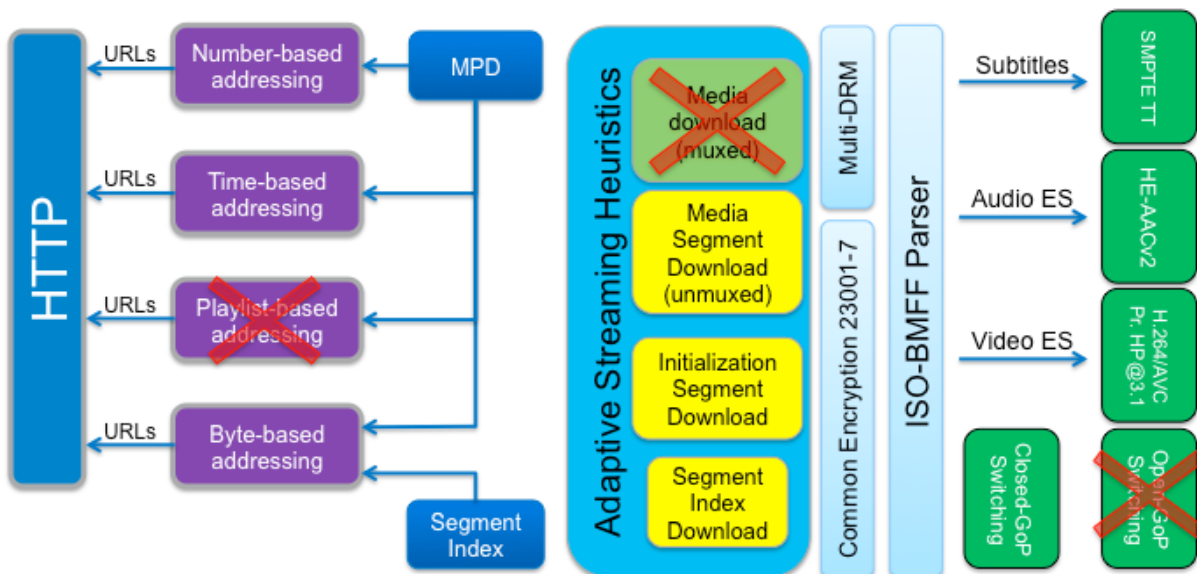


Figure 2 DASH aspects of a DASH-AVC/264 client compared to a client supporting the union of DASH ISO BMFF live and on-demand profile.

---

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

- 3 • support of HTTP GET and partial GET requests to download Segments and Subsegments
- 4 • three different addressing schemes: number and time-based templating as well as byte  
5 range based requests.
- 6 • support of metadata as provided in the MPD and Segment Index
- 7 • download of Media Segments, Initialization Segments and Segment Index
- 8 • ISO BMFF parsing
- 9 • synchronized presentation of media components from different Adaptation Sets
- 10 • switching of video streams at closed GOP boundaries

### 11 **3.3.3. Seamless switching**

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

21 The formats defined in section 3.2 provide suitable switching opportunities at (sub)segment bound-  
22 aries.

### 23 **3.3.4. DASH Client Requirements**

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

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

## 35 **3.4. Transport and Protocol-Related Issues**

### 36 **3.4.1. General**

37 Servers and clients operating in the context of the interoperability points defined in this document  
38 shall support the normative parts of HTTP/1.1 as defined in RFC 7230 [22], RFC 7231 [23], RFC  
39 7232 [24], RFC 7233 [25], and RFC 7234 [26].

40 Specific requirements and recommendations are provided below.

41 Note: IETF recently obsoleted RFC 2616 and replaced it with the six RFCs referred above.

42 The changes are generally text clarifications and in some cases, additional constraints to

---

1 address security or interoperability issues. Each new RFC contains details of the changes  
2 compared to RFC2616. The IETF strongly recommends to reference and use the new RFCs  
3 that collectively replace RFC2616. This version of DASH-IF IOP addresses this aspect.  
4

5 MPEG-DASH explicitly permits the use of https as a scheme and hence, HTTP over TLS as a  
6 transport protocol as defined in RFC 2818 [66]. For more details refer to section 7.2.

### 7 **3.4.2. Server Requirements and Guidelines**

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

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

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

15 **BaseURL**@byteRange="\$base\$?\$query\$&range=\$first\$-\$last\$"

### 16 **3.4.3. Client Requirements and Guidelines**

17 Clients shall support byte range requests, i.e. issue partial GETs to subsegments as defined in RFC  
18 7233 [25]. Range requests may also be issued by using Annex E of 23009-1 using the syntax of  
19 the second example in Annex E.3,

20 **BaseURL**@byteRange="\$base\$?\$query\$&range=\$first\$-\$last\$"

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

23 Clients should support the normative aspects of the HTTP state management mechanisms (also  
24 known as Cookies) as defined in RFC 6265 [42] for first-party cookies.

### 25 **3.4.4. Transforming Proxies and Other Adaptation Middleboxes**

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

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

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

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

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

---

## 1 3.5. Synchronization Considerations

2 In order to properly access MPDs and Segments that are available on DASH servers, DASH serv-  
3 ers and clients should synchronize their clocks to a globally accurate time standard. Specifically it  
4 is expected that the Segment Availability Times as computed from the MPD according to ISO/IEC  
5 23009-1 [5], section 5.3.9.5 and additional details in ISO/IEC 23009-3 [7], section 6.4 are accu-  
6 rately announced in the MPD.

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

- 8 • Usage of NTP or SNTP as defined in RFC5905 [41].
- 9 • The Date general-header field in the HTTP header (see RFC 7231 [23] section 7.1.1.2)  
10 represents the date and time at which the message was originated, and may be used as an  
11 indication of the actual time.

12 Anticipated inaccuracy of the timing source should be taken into account when requesting seg-  
13 ments close to their segment availability time boundaries.

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

## 15 3.6. Considerations for Live Services

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

## 17 3.7. Considerations on Ad Insertion

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

## 20 3.8. Switching across Adaptation Sets

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

28 A content author may signal such seamless switching property across Adaptation Sets by providing  
29 a Supplemental Descriptor along with an Adaptation Set with `@schemeIdURI` set to  
30 `http://dashif.org/descriptor/AdaptationSetSwitching` and the `@value` is a  
31 comma-separated list of Adaptation Set IDs that may be seamlessly switched to from this Adapta-  
32 tion Set.

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

37 As an example, a content author may signal that seamless switching across an H.264/AVC Adap-  
38 tation Set with `AdaptationSet@id="h264"` and an HEVC Adaptation Set with `Adapta-`  
39 `tionSet@id="h265"` is possible by adding a Supplemental Descriptor to the H.264/AVC Ad-

1 aptation Set with @schemeIdURI set to `http://dashif.org/descriptor/Adapta-`  
 2 `tionSetSwitching` and the @value="h265" and by adding a Supplemental Descriptor to  
 3 the HEVC Adaptation Set with @schemeIdURI set to `http://dashif.org/de-`  
 4 `scriptor/AdaptationSetSwitching` and the @value="h264".

5 In addition, if the content author signals the ability of Adaptation Set switching for any Adaptation  
 6 Sets

- 7 - with @contentType="video" then the parameters as defined in section 3.2.4 for an  
 8 Adaption Set shall also hold for all Adaptation Sets that are included in the @value at-  
 9 tribute.
- 10 - with @contentType="audio" then the parameters as defined in section 3.2.4 for an  
 11 Adaption Set shall also hold for all Adaptation Sets that are included in the @value at-  
 12 tribute.

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

## 16 4. Live Services

### 17 4.1. Introduction

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

20 For this purpose, this section

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

27 The main features and differences of these two modes are provided in the following Table 3:

28 **Table 3 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 pe- riod	by Inband Event messages in the segments.
URL generation	based on MPD	based on MPD and segment infor- mation
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

29



---

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

## 3 **4.2. Overview Dynamic and Live Media Presentations**

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

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

11 Dynamic services may be used for different types of services:

- 12 1. **Dynamic Distribution of Available Content:** Services, for which content is made avail-  
13 able as dynamic content, but the content is entirely generated prior to distribution. In this  
14 case the details of the Media Presentation, especially the Segments (duration, URLs) are  
15 known and can be announced in a single MPD without MPD updates. This addresses use  
16 cases 2 and 3 in Annex B.
- 17 2. **MPD-controlled Live Service:** Services for which the content is typically generated on  
18 the fly, and the MPD needs to be updated occasionally to reflect changes in the service  
19 offerings. For such a service, the DASH client operates solely on information in the  
20 MPD. This addresses the use cases 4 and 5 in Annex B.
- 21 3. **MPD and Segment-controlled Live:** Services for which the content is typically gener-  
22 ated on the fly, and the MPD may need to be updated on short notice to reflect changes in  
23 the service offerings. For such a service, the DASH client operates on information in the  
24 MPD and is expected to parse segments to extract relevant information for proper opera-  
25 tion. This addresses the use cases 4 and 5, but also takes into account the advanced use  
26 cases.

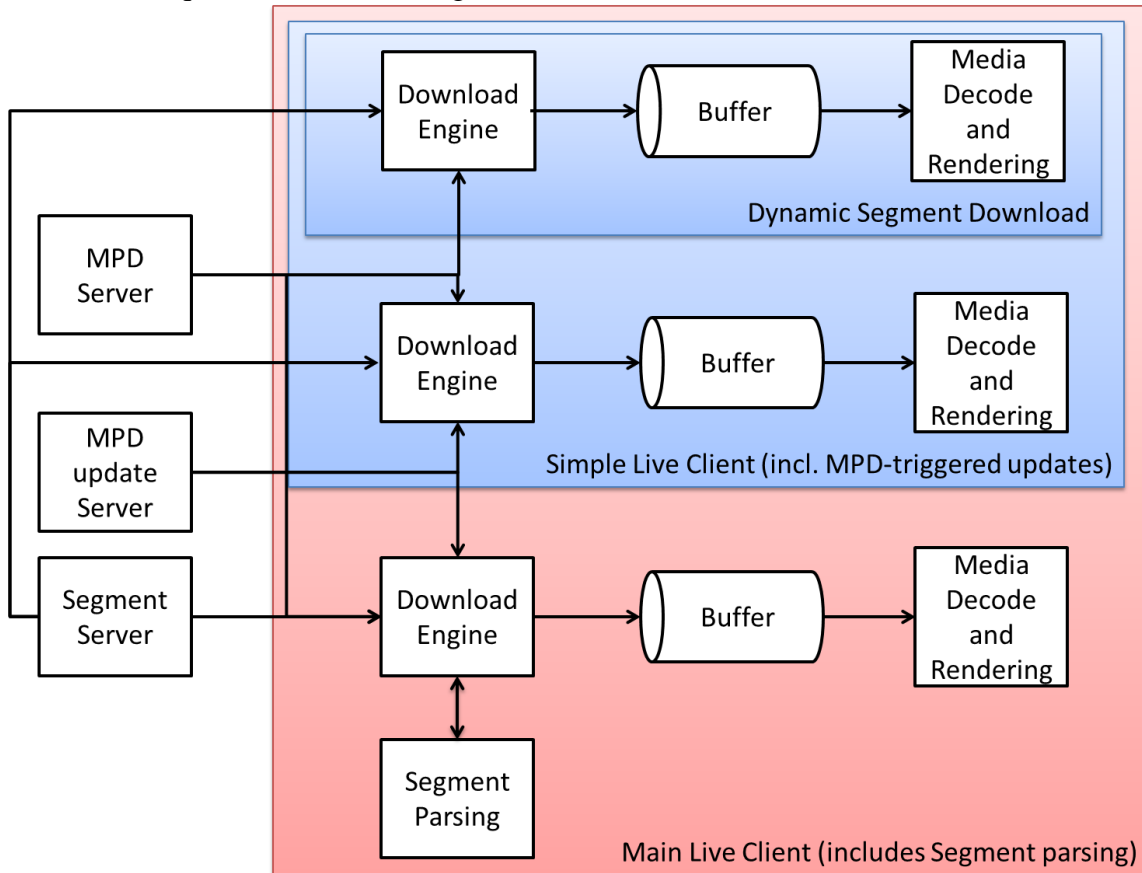
27 Dynamic and Live services are typically controlled by different client transactions and server-side  
28 signaling.

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

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

40 The MPD may be updated on the server based on certain rules and clients consuming the service  
41 are expected to update MPDs based on certain triggers. The triggers may be provided by the MPD

1 itself or by information included in Segments. Depending on the service offering, different client  
 2 operations are required as shown in Figure 3.



3  
 4 **Figure 3 Different Client Models**

5 The basic functions a live clients describes in this document are as follows:

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

---

1 Segments if the service offering provides this feature. MPDs are refetched and revali-  
2 dated when the currently available MPD expires based on expiry information in the Seg-  
3 ments.

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

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

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

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

12 Requirements and recommendations for client-server timing synchronization are defined in sec-  
13 tion 4.7.

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

16 Interoperability Aspects are defined in section 4.9.

## 17 **4.3. Dynamic Segment Download**

### 18 **4.3.1. Background and Assumptions**

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

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

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

### 28 **4.3.2. Preliminaries**

#### 29 **4.3.2.1. MPD Information**

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

35 For a dynamic service the MPD should at least contain information as available in Table 4. Infor-  
36 mation included there may be used to compute a list of announced Segments, Segment Availability  
37 Times and URLs.

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

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

7 **Table 4 -- Information related to Segment Information and Availability Times for a dynamic service**

<b>MPD Information</b>	<b>Status</b>	<b>Comment</b>
<b>MPD</b> @type	mandatory, set to "dynamic"	the type of the Media Presentation is dynamic, i.e. Segments get available over time.
<b>MPD</b> @availabilityStartTime	mandatory	the start time is the anchor for the MPD in wall-clock time. The value is denoted as <i>AST</i> in the following.
<b>MPD</b> @mediaPresentationDuration	mandatory (for the considered use cases)	provides the duration of the Media Presentation.
<b>MPD</b> @suggestedPresentationDelay	optional, but recommended	suggested presentation delay as delta to segment availability start time. The value is denoted as <i>SPD</i> . Details on the setting and usage of the parameter is provided in the following.
<b>MPD</b> @minBufferTime	mandatory	minimum buffer time, used in conjunction with the @bandwidth attribute of each Representation. The value is denoted as <i>MBT</i> . Details on the setting and usage of the parameter is provided in the following.
<b>MPD</b> @timeShiftBufferDepth	optional, but recommended	time shift buffer depth of the media presentation. The value is denoted as <i>TSB</i> . Details on the setting and usage of the parameter is provided in the following.
<b>Period</b> @start	Mandatory for the first Period in the MPD	the start time of the Period relative to the MPD availability start time.
<b>Representation</b> @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 <b>MPD@availabilityStartTime</b>
<b>SegmentTemplate@media</b>	mandatory	The template for the Media Segment assigned to a Representation.
<b>SegmentTemplate@startNumber</b>	optional default	number of the first segment in the Period assigned to a Representation
<b>SegmentTemplate@timescale</b>	optional default	timescale for this Representation.
<b>SegmentTemplate@duration</b> <b>SegmentTemplate.SegmentTimeline</b>	exactly one of <b>SegmentTemplate@duration</b> or <b>SegmentTemplate.SegmentTimeline</b> must be present per Representation.	the duration of each Segment in units of a time.

### 1 4.3.2.2. Segment Information Derivation

#### 2 4.3.2.2.1. Introduction

3 Based on an MPD including information as documented in Table 4 and available at time *NOW* on  
4 the server, a synchronized DASH client derives the information of the list of Segments for each  
5 Representation in each Period. This section only describes the information that is expressed by the  
6 values in the MPD. The generation of the information on the server and the usage of the infor-  
7 mation in the client is discussed in section 4.3.3 and 4.3.4, respectively.

8 MPD information is provided in subsection 4.3.2.2.3. The Period based information is documented  
9 in sub-section 4.3.2.2.4, and the Representation information is documented in sub-section  
10 4.3.2.2.5.

#### 11 4.3.2.2.2. Definitions

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

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

#### 20 4.3.2.2.3. MPD Information

21 For a dynamic service without MPD updates, the following information shall be present and not  
22 present in the MPD (also please refer to Table 4):

- 23 • The **MPD@type** shall be set to "dynamic".
- 24 • The **MPD@mediaPresentationDuration** shall be present, or the **Period@duration** of the last Period shall be present.
- 25 • The **MPD@minimumUpdatePeriod** shall not be present.

---

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

#### 3 **4.3.2.2.4. Period Information**

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

7 Each regular Period  $i$  in the MPD is assigned a

- 8 • Period start time  $PSwc[i]$  in wall-clock time,
- 9 • Period end time  $PEwc[i]$ , in wall-clock time.

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

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

- 13 • If the attribute `@start` is present in the **Period**, then  $PSwc[i]$  is the sum of *AST* and the  
14 value of this attribute.
- 15 • If the `@start` attribute is absent, but the previous **Period** element contains a `@dura-`  
16 `tion` attribute then the start time of the Period is the sum of the start time of the previous  
17 Period  $PSwc[i]$  and the value of the attribute `@duration` of the previous Period. Note  
18 that if both are present, then the `@start` of the new Period takes precedence over the  
19 information derived from the `@duration` attribute.

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

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

#### 28 **4.3.2.2.5. Representation Information**

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

31 If the **SegmentTemplate.SegmentTimeline** is present and the **SegmentTem-**  
32 **plate@duration** is not present, the **SegmentTimeline** element contains  $N_S$  **S** elements  
33 indexed with  $s=1, \dots, N_S$ , then let

- 34 •  $ts$  the value of the `@timescale` attribute
- 35 •  $ato$  is the value of the `@availabilityTimeOffset` attribute, if present. Otherwise it  
36 is zero.
- 37 •  $t[s]$  be the value of `@t` of the  $s$ -th **S** element,
- 38 •  $d[s]$  be the value of `@d` of the  $s$ -th **S** element
- 39 •  $r[s]$  be,

- 
- 1       ○ if the @r value is greater than or equal to zero
  - 2           ▪ one more than the value of @r of the s-th S element. Note that if @r is
  - 3           smaller than the end of this segment timeline element, then this Representa-
  - 4           tion contains gaps and no media is present for this gap.
  - 5       ○ else
  - 6           ▪ if  $t[s+1]$  is present, then  $r[s]$  is the ceil of  $(t[s+1] - t[s])/d[s]$
  - 7           ▪ else  $r[s]$  is the ceil of  $(PEwc[i] - PSwc[i] - t[s]/ts)*ts/d[s]$

8 If the **SegmentTemplate**@duration is present and the **SegmentTemplate.Seg-**  
 9 **mentTimeline** is not present, then

- 10       •  $N_s=1$ ,
- 11       • *ato* is the value of the @availabilityTimeOffset attribute, if present. Otherwise it
- 12       is zero.
- 13       • *ts* the value of the @timescale attribute
- 14       •  $t[s]$  is 0,
- 15       • the  $d[s]$  is the value of @duration attribute
- 16       •  $r[s]$  is the ceil of  $(PEwc[i] - PSwc[i] - t[s]/ts)*ts/d[s]$

#### 17 4.3.2.2.6. Media Time Information of Segment

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

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

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

24

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

#### 26 4.3.2.2.7. Segment List Parameters

27 For each Period  $i$  with Period start time  $PSwc[i]$  and Period end time  $PEwc[i]$  and each Representa-  
 28 tion  $r$  in the Period the following information can be computed:

- 29       • the presentation time offset described in the MPD,  $o[i,r]$
- 30       • the availability time offset of this Representation,  $ato[r]$
- 31       • the number of the first segment described in the MPD,  $k1[i,r]$
- 32       • the number of the last segment described in the MPD,  $k2[i,r]$
- 33       • segment availability start time of the initialization segment  $SAST[0,i,r]$
- 34       • segment availability end time of the initialization segment  $SAET[0,i,r]$
- 35       • segment availability start time of each media segment  $SAST[k,i,r]$ ,  $k=k1, \dots, k2$
- 36       • segment availability end time of each media segment  $SAET[k,i,r]$ ,  $k=k1, \dots, k2$
- 37       • adjusted segment availability start time  $ASAST[0,i,r]$ ,  $k=0, k1, \dots, k2$
- 38       • segment duration of each media segment  $SD[k,i,r]$ ,  $k=k1, \dots, k2$

---

1       • the URL of each of the segments,  $URL[k,i,r]$

2 In addition,

3       • the latest available Period  $i[NOW]$  and the latest segment available at the server  $k[NOW]$

4       can be computed. This segment is also referred to as *live edge segment*.

5       • the earliest available Period  $i^*[NOW]$  and the earliest segment available at the server

6        $k^*[NOW]$  can be computed.

7 Based on the above information, for each Representation  $r$  in a Period  $i$ , the segment availability

8 start time  $SAST[k,i,r]$ , the segment availability end time of each segment  $SAET[k,i,r]$ , the segment

9 duration of each segment  $SD[k,i,r]$ , and the URL of each of the segments,  $URL[k,i,r]$  within one

10 Period  $i$  be derived as follows using the URL Template function  $URLTemplate(Replace-$

11  $mentString, Address)$  as documented in subsection 4.3.2.2.8:

12       •  $k=0$

13       •  $SAST[0,i,r] = PSwc[i]$

14       •  $ASAST[0,i,r] = PSwc[i] - ato$

15       • for  $s=1, \dots N_s [i,r]$

16           ○  $k = k + 1$

17           ○  $SAST[k,i,r] = PSwc[i] + (t[s,i,r] + d[s,i,r] - o[i,r])/ts$

18           ○  $ASAST[k,i,r] = SAST[k,i,r] - ato$

19           ○  $SD[k,i,r] = d[s,i,r]/ts$

20           ○  $SAET[k,i,r] = SAST[k,i,r] + TSB + d[s,i,r]/ts$

21           ○ if **SegmentTemplate**@media contains \$Number\$

22               ▪  $Address=@startNumber$

23               ▪  $URL[k,i,r] = URLTemplate(\$Number\$, Address)$

24           else

25               ▪  $Address = t[s,i,r]$

26               ▪  $URL[k,i,r] = URLTemplate(\$Time\$, Address)$

27           ○ for  $j = 1, \dots, r[s,i,r]$

28               ▪  $k = k + 1$

29               ▪  $SAST[k,i,r] = SAST[k-1,i,r] + d[s,i,r]/ts$

30               ▪  $ASAST[k,i,r] = SAST[k,i,r] - ato$

31

32               ▪  $SAET[k,i,r] = SAST[k,i,r] + TSB + d[s,i,r] /ts$

33

34               ▪  $SD[k,i,r] = d[s,i,r] /ts$

35               ▪ if **SegmentTemplate**@media contains \$Number\$

36                   ▪  $Address = Address + 1$

37                   ▪  $URL[k,i,r] = URLTemplate(\$Number\$, Address)$

38           else

39               ▪  $Address = Address + d[s,i,r]$

40               ▪  $URL[k,i,r] = URLTemplate(\$Time\$, Address)$

41       •  $k2[i,r] = k$

42       •  $SAET[0,i,r] = SAET[k2[i,r],i,r]$



---

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

3 Hence, the number of the first media segment described in the MPD for this Period,  $k_1[i,r]$ , is the  
4 smallest  $k=1, 2, \dots$  for which  $SAST[k,i,r] \geq NOW$ .

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

7 The latest available segment  $k[NOW]$  available for a Representation of Period  $i[NOW]$  (also the  
8 live edge segment) is the segment with the largest  $k=0,1,2,\dots$  such that  $SAST[k,i,r]$  is smaller than  
9 or equal to *NOW*. Note that this contains the Initialization Segment with  $k=0$  as not necessarily any  
10 media segment may yet be available for Period  $i[NOW]$ . In this case, last media segment  
11  $k_2[i[NOW]-1,r]$ , i.e., the last media segment of the previous Period is the latest accessible media  
12 Segment.

13 However, if the `@availabilityTimeOffset` is present, then the segments for this Representa-  
14 tion are available earlier than the nominal segment availability start time, namely at  $ASAST[k,i,r]$ .

#### 15 **4.3.2.2.8. URL Generation with Segment Template**

16 The function URL Template function `URLTemplate(ReplacementString, Address)` gen-  
17 erates a URL. For details refer to ISO/IEC 23009-1 [1], section 5.3.9.4. Once the Segment is gen-  
18 erated, processing of the Base URLs that apply on this segment level is done as defined in ISO/IEC  
19 23009-1, section 5.6.

#### 20 **4.3.2.2.9. Synchronized Playout and Seamless Switching**

21 In order to achieve synchronized playout across different Representations, typically from different  
22 Adaptation Sets, the different Representations are synchronized according to the presentation time  
23 in the Period. Specifically, the earliest presentation time of each Segment according to section  
24 4.3.2.2.6 determines the playout of the Segment in the Period and therefore enables synchronized  
25 playout of different media components as well as seamless switching within one media component.

### 26 **4.3.3. Service Offering Requirements and Guidelines**

#### 27 **4.3.3.1. General Service Offering Requirements**

28 For dynamic service offerings, the MPD shall conform to DASH-IF IOP as defined in section 3  
29 and shall at least contain the mandatory information as documented in Table 4.

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

- 31 • all Segments for all Representations in all Periods as announced in an MPD shall be  
32 available latest at the announced segment availability start time  $SAST[k,i,r]$  at all  
33  $URL[k,i,r]$  as derived in section 4.3.2.2;
- 34 • all Segments for all Representations in all Periods as announced in an MPD shall at least  
35 be available until the announced segment availability end time  $SAET[k,i,r]$  at all  
36  $URL[k,i,r]$  as derived in section 4.3.2.2;
- 37 • for all Media Segments for all Representations in all Periods as announced in an MPD the  
38 Segment in this Period is available prior to the sum of Period start, earliest presentation  
39 time and segment duration, i.e.  $SAST[k,i,r] \leq PSwc[i] + SD[k,r,i] + EPT[k,r,i]$ ;

- 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 5 – Basic Service Offering**

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

- $k1 = 1$
- $k2 = \text{ceil}(PDURATION/SDURATION)$
- for  $k = 1, \dots, k2$ 
  - $SAST[k] = START + PSTART + k*SDURATION$

- 
- 1           ○ SAET[k] = SAST[k] + TSB + SDURATION
  - 2           ○ SD[k] = SDURATION
  - 3           ○ URL[k] = http://example.com/\$RepresentationID\$/k
  - 4       • The segment availability times of the Initialization Segment are as follows:
  - 5           ○ SAST[0] = START + PSTART
  - 6           ○ SAET[0] = SAET[k2]

#### 7 4.3.3.2.2. Basic Parameter Settings

8 In the following recommendations are provided for the

- 9       • Time Shift Buffer Depth (TSB):
  - 10           ○ If the content should be consumed at the live edge, then the time shift buffer depth
  - 11           should be set short. However, the TSB should not be smaller than the recommended
  - 12           value of 4\*SDURATION and 6 seconds in media time in order for the client to do
  - 13           some prebuffering in more difficult network conditions.
  - 14           ○ If no restrictions on the accessibility of the content are provided, then the TSB may
  - 15           be set to a large value that even exceeds PDURATION.
- 16       • Suggested Presentation Delay (SPD)
  - 17           ○ If synchronized play-out with other devices adhering to the same rule is desired
  - 18           and/or the service provider wants to define the typical live edge of the program,
  - 19           then this value should be provided. The service provider should set the value taking
  - 20           into account at least the following:
    - 21               ▪ the desired end-to-end latency
    - 22               ▪ the typical required buffering in the client, for example based on the net-
    - 23               work condition
    - 24               ▪ the segment duration SDURATION
    - 25               ▪ the time shift buffer depth TSB
  - 26           ○ A reasonable value may be 2 to 4 times of the segment duration SDURATION, but
  - 27           the time should not be smaller than 4 seconds in order for the client to maintain
  - 28           some buffering.
- 29       • Segment Duration (SDURATION)
  - 30           ○ The segment duration typically influences the end-to-end latency, but also the
  - 31           switching and random access granularity as in DASH-264/AVC each segment
  - 32           starts with a stream access point which can also be used as a switch point. The
  - 33           service provider should set the value taking into account at least the following:
    - 34               ▪ the desired end-to-end latency
    - 35               ▪ the desired compression efficiency
    - 36               ▪ the start-up latency
    - 37               ▪ the desired switching granularity
    - 38               ▪ the desired amount of HTTP requests per second
    - 39               ▪ the variability of the expected network conditions
  - 40           ○ Reasonable values for segment durations are between 1 second and 10 seconds.
- 41       • Minimum Buffer Time (MBT) and bandwidth (BW)
  - 42           ○ the value of the minimum buffer time **does not provide any instructions to the**
  - 43           **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 8.

**Table 6 – Basic Service Offering**

MPD Information	Value
-----------------	-------

<b>MPD@type</b>	dynamic
<b>MPD@availabilityStartTime</b>	START
<b>MPD@mediaPresentationDuration</b>	43sec
<b>MPD@suggestedPresentationDelay</b>	15sec
<b>MPD@minBufferTime</b>	5sec
<b>MPD@timeShiftBufferDepth</b>	25sec
<b>MPD.BaseURL</b>	"http://example.com/"
<b>Period@start</b>	0
<b>SegmentTemplate@media</b>	"\$RepresentationID\$/\$Number\$"
<b>SegmentTemplate@startNumber</b>	1
<b>SegmentTemplate@duration</b>	5sec

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

- 2
- $k_1 = 1, k_2 = 9$
  - for  $k = 1, \dots, k_2$ 
    - $SAST[k] = START + k * 5sec$
    - $SAET[k] = SAST[k] + 30sec$
    - $URL[k] = http://example.com/1/k$
  - The segment availability times of the Initialization Segment are as follows:
    - $SAST[0] = START$
    - $SAET[0] = START + 75 sec$
- 3
- 4
- 5
- 6
- 7
- 8
- 9

10 Figure 4 shows the availability of segments on the *server* for different times *NOW*. In particular,  
 11 before *START* no segment is available, but the segment URLs are valid. With time *NOW* advanc-  
 12 ing, segments get available.

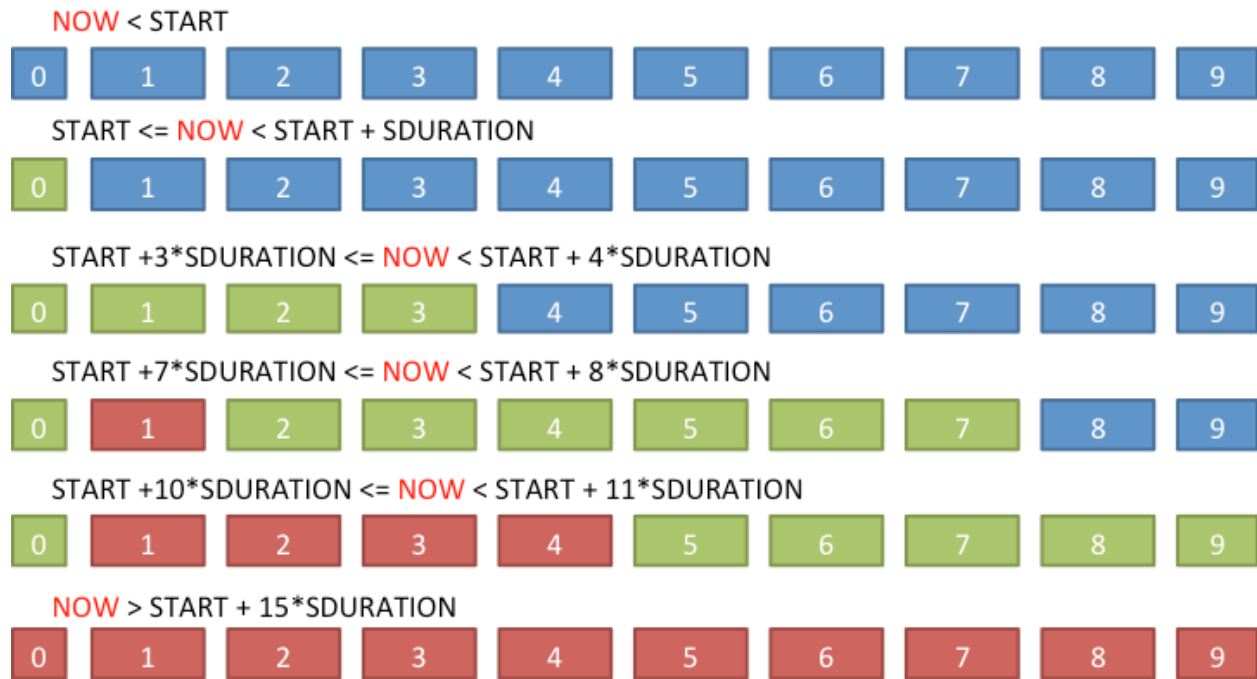


Figure 4 Segment Availability on the Server for different time NOW (blue = valid but not yet available segment, green = available Segment, red = unavailable Segment)

#### 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 7. 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 7 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

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

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

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

3 • Period 1

- 4 ○  $PSwc[1] = START + PSTART$
- 5 ○  $PEwc[1] = START + PSTART2$
- 6 ○  $k1 = 1$
- 7 ○  $k2 = \text{ceil}((PSTART2 - PSTART1) / SDURATION)$
- 8 ○ for  $k = 1, \dots, k2$ 
  - 9 ▪  $SAST[k] = PSwc[1] + k * SDURATION$
  - 10 ▪  $SAET[k] = SAST[k] + TSB + SDURATION$
  - 11 ▪  $SD[k] = SDURATION$
  - 12 ▪  $URL[k] = \text{http://example.com/1}/ \$RepresentationID$/k$
- 13 ○  $SAST[0] = PSwc[1]$
- 14 ○  $SAET[0] = SAET[k2]$

15 • Period 2

- 16 ○  $PSwc[2] = START + PSTART2$
- 17 ○  $PEwc[2] = START + PSTART3$
- 18 ○  $k1 = 1$
- 19 ○  $k2 = \text{ceil}((PSTART3 - PSTART2) / SDURATION2)$
- 20 ○ for  $k = 1, \dots, k2$ 
  - 21 ▪  $SAST[k] = PSwc[2] + k * SDURATION2$
  - 22 ▪  $ASAST[k] = SAST[k] - ATO2$
  - 23 ▪  $SAET[k] = SAST[k] + TSB + SDURATION2$
  - 24 ▪  $SD[k] = SDURATION2$
  - 25 ▪  $URL[k] = \text{http://example.com/2}/ \$RepresentationID$/k$
- 26 ○  $SAST[0] = PSwc[2]$
- 27 ○  $SAET[0] = SAET[k2]$

28 • Period 3

- 29 ○  $PSwc[3] = START + PSTART3$
- 30 ○  $PEwc[3] = START + PDURATION$
- 31 ○  $k1 = 1$
- 32 ○  $k2 = \text{ceil}((PSTART3 - PDURATION) / SDURATION1)$

- 
- 1       ○ for  $k = 1, \dots, k_2$
  - 2           ▪  $SAST[k] = PSwc[3] + k * SDURATION1$
  - 3           ▪  $SAET[k] = SAST[k] + TSB + SDURATION1$
  - 4           ▪  $SD[k] = SDURATION1$
  - 5           ▪  $URL[k] = "http://example.com/1/\$RepresentationID\$/ (k+STARTNUMBER2-1) "$
  - 6
  - 7       ○  $SAST[0] = PSwc[3]$
  - 8       ○  $SAET[0] = SAET[k_2]$

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

#### 11 4.3.3.3.2. Continuous Period Offering

12 Note: This is aligned with Amd.3 of ISO/IEC 23009-1:2014 [5] and may be referenced in  
13 a future version of this document.

14 In certain circumstances the Content Provider offers content in the next Period that is a continua-  
15 tion of the content in the previous Period, possibly in the immediately following Period or in a  
16 later Period. The latter case applies for example after an advertisement Period had been inserted.  
17 The content provider may express that the media components contained in two Adaptation Sets in  
18 two different Periods are *associated* by assigning equivalent Asset Identifiers to both Periods and  
19 by identifying both Adaptation Sets with identical value for the attribute @id.

20 If Adaptation Sets in two different Periods are *associated*, then the Adaptation Set parameters  
21 defined in ISO/IEC 23009-1, section 5.3.3.1, must be identical for the two Adaptation Sets.

22 Furthermore, two Adaptation Sets in one MPD are *period-continuous* if all of the following holds:

- 23       • The Adaptation Sets are associated.
- 24       • The @presentationTimeOffset is present or can be inferred as 0 for all Representations in both Adaptation Sets.
- 25       • Within one Adaptation Set, the value of @presentationTimeOffset is identical for all Representations.
- 26       • 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 other Adaptation Set.
- 27       • 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 the Representation.
- 28
- 29
- 30
- 31
- 32
- 33

34 Content authors should signal *period-continuous* Adaptation Sets by signalling the presentation  
35 duration. The *presentation duration* of a Representation is the difference between the end presen-  
36 tation time of the Representation and the earliest presentation time of the Representation. The  
37 presentation time duration has the same unit as presentation time offset, i.e. @timescale, and  
38 expresses the exact presentation duration of the Representation.

39 The presentation duration may be signalled by

- 40       - A supplemental descriptor with @scheme\_id\_URI set to "urn:mpeg:dash:pe-  
41       riod\_continuity:2014" may be provided for an Adaptation Set with



- 1           ○ the @value of the descriptor, PID, matching the value of an @id of a Period that
- 2           is contained in the MPD,
- 3           ○ the value of the **AdaptationSet@id** being AID,
- 4           ○ the value of the @presentationTimeOffset for this Adaptation Set is pro-
- 5           vided and is PTO.

6 If this signal is present, then for the Period with the value of the **Period@id** being PID and for  
7 the Adaptation Set with **AdaptationSet@id** being AID, the presentation duration of each Rep-  
8 resentation in this Adaptation Set is obtained as the difference of PTO minus the value of the  
9 @presentationTimeOffset.

10 Content authors should offer an MPD with *period-continuous* Adaptation Sets if the MPD contains  
11 Periods with identical Asset Identifiers.

#### 12 **4.3.3.4. Content Offering with Segment Timeline**

##### 13 **4.3.3.4.1. Basic Operation**

14 In order to offer a dynamic service that takes into account

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

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

19 **Table 8 – Service Offering with Segment Timeline**

MPD Information	Value
<b>MPD@type</b>	dynamic
<b>MPD@availabilityStartTime</b>	START
<b>MPD@mediaPresentationDuration</b>	PDURATION
<b>MPD@suggestedPresentationDelay</b>	SPD
<b>MPD@minBufferTime</b>	MBT
<b>MPD@timeShiftBufferDepth</b>	TSB
<b>MPD.BaseURL</b>	"http://example.com/"
<b>Period@start</b>	PSTART
<b>SegmentTemplate@media</b>	"\$RepresentationID\$/ \$Num- ber\$"
<b>SegmentTemplate@startNumber</b>	1
<b>SegmentTemplate.SegmentTimeline</b>	t[i], n[i], d[i], r[i]

20 According to the work-flow shown in Annex B:

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

---

1 offer occasional shorter segments for encoding optimizations, e.g. at scene changes, or  
2 segment gaps (for details see section 6). If such an irregular segment is published the MPD  
3 needs to document this by a new **S** element in the segment timeline.

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

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

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

#### 12 **4.3.3.4.2. Basic Parameter Settings**

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

#### 16 **4.3.3.5. Joining Recommendation**

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

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

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

- 25 • the `t` parameter, or
- 26 • both the `period` and `t` parameter, in the MPD URL provided to the DASH client, or
- 27 • the `utc` parameter, for details refer to Amd.3 of ISO/IEC 23009-1:2014 [5].

28 The format and behaviour of MPD Anchors is defined in section C.4 of ISO/IEC 23009-1. Specif-  
29 ically the `utc` parameter is defined in Amd.3 of ISO/IEC 23009-1:2014 [5].

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

```
32 #t=0
```

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

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

36 Notes:

- 37 • as per section C.4 of ISO/IEC 23009-1 the time indicated using the `t` parameter is as per  
38 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 5.

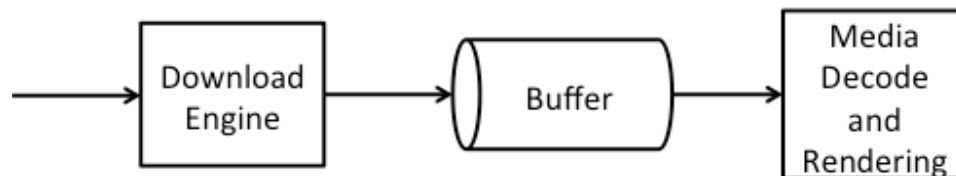


Figure 5 Simple Client Model

Assume that the client has access to an MPD and the MPD contains the parameters in Table 4, 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 4 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 5 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

---

1 the presentation time in the Period as documented in section 4.3.2.2.9. For synchronized  
2 playout, the exact presentation times in the media shall be used.

3 Once presentation has started, the playout process is continuous. The playout process ex-  
4 pects media to be present in the buffer continuously. If the **MPD@suggestedPresent-**  
5 **tationDelay** is present, then this value may be used as the presentation delay *PD*. If  
6 the **MPD@suggestedPresentationDelay** is not present, but the client is expected  
7 to consume the service at the live edge, then a suitable presentation delay should be se-  
8 lected, typically between the value of **@minBufferTime** and the value of  
9 **@timeShiftBufferDepth**. It is recommended that the client starts rendering the first  
10 sample of the downloaded media segment *k* with earliest presentation time  $EPT(k)$  at  
11  $PSwc[i] + (EPT(k) - o[r,i]) + PD$ . For details on selecting and minimizing end-to-end la-  
12 tency as well as the start-up latency, see section 4.3.4.4.

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

#### 29 4.3.4.2. Determining the Segment List

30 For a single Period content the client determines the available Segment List at time *NOW* according  
31 to section 4.3.2.2.7 taking into account the simplified offering in Table 5 as

- 32 •  $k1 = 1$
- 33 •  $k2 = \text{ceil}(PDURATION/SDURATION)$
- 34 •  $SAST[k] = START + PSTART + k*SDURATION$  for  $k = 0, 1, \dots, k2$
- 35 •  $ASAST[k] = SAST[k] - ATO$
- 36 •  $SAET[k] = SAST[k] + TSB + SDURATION$  for  $k = 1, \dots, k2$
- 37 •  $SAET[0] = SAET[k2]$
- 38 •  $SD[k] = SDURATION$
- 39 •  $URL[k] = \text{http}://\text{example.com}/\text{\$RepresentationID\$/k}$
- 40 •  $k[NOW] = \text{MIN}(\text{floor}((NOW - START - PSTART)/SDURATION), k2)$
- 41 •  $k*[NOW] = \text{MAX}(k1, \text{floor}((NOW - START - PSTART - TSB)/SDURATION))$

---

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

#### 4 4.3.4.3. Multi-Period Content

5 In an extension to the description in section 4.3.4.1 assume now that the client has access to an  
6 MPD and the MPD contains content with multiple Periods, for example following the parameters  
7 in Table 7. The start time of each Period is computed as period start time  $PSwc[i]$ . and the MPD-  
8 URL does not include any fragment parameters according to section 4.3.3.5.

9 In an extension of bullet 3 in section 4.3.4.1,

10 the client creates a list of accessible Segments at least for each selected Representation  
11 taking into account the information in the MPD as documented in Table 4 and the current  
12 time  $NOW$  in the client and in particular the segment closest to the live edge referred to the  
13 *live edge segment*.

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

- 16 • if  $NOW \leq PSwc[1]$ 
  - 17 ○ no segment is yet available
- 18 • else if  $NOW > PSwc[i^*]$ 
  - 19 ○ the last one and the latest segment is available is  $k2[i^*]$
- 20 • else if  $NOW > PSwc[i^*] + TSB$ 
  - 21 ○ no segment is available any more
- 22 • else if  $PSwc[1] < NOW \leq PEwc[i^*]$ 
  - 23 ▪  $i'$  the such that  $PSwc[i'] < NOW \leq PEwc[i']$
  - 24 ▪  $k[NOW] = \text{MIN}(\text{floor}((NOW - PEwc[i'] - PSwc[i'])/SDURATION[i']),$   
25  $k2)$
- 26 • Note again that if  $k[NOW]$  is 0, then only the Initialization Segment is available. If  
27 the Period is not the first one, then the last available Media Segment is the last  
28 Media Segment of the previous Period.

29 In an extension of bullet 9 in section 4.3.4.1,

30 the client consumes media in one Period. Once the client is consuming media contained in  
31 the Segments towards the end of the announced media in the Representation, and the Rep-  
32 resentation is contained not in the last Period, then the DASH clients generally needs to  
33 reselect the Adaptation Sets and a Representation in same manner as described in bullet 1  
34 and 2 in section 4.3.4.1. Also steps 3, 4, 5 and 6 need to be carried out at the transition of  
35 a Period. Generally, audio/video switching across period boundaries may not be seamless.  
36 According to ISO/IEC 23009-1, section 7.2.1, at the start of a new Period, the playout  
37 procedure of the media content components may need to be adjusted at the end of the pre-  
38 ceding Period to match the *PeriodStart* time of the new Period as there may be small over-  
39 laps or gaps with the Representation at the end of the preceding Period. Overlaps (respec-  
40 tively gaps) may result from Media Segments with actual presentation duration of the me-  
41 dia stream longer (respectively shorter) than indicated by the Period duration. Also in the

---

1 beginning of a Period, if the earliest presentation time of any access unit of a Representa-  
2 tion is not equal to the presentation time offset signalled in the @presentation-  
3 TimeOffset, then the playout procedures need to be adjusted accordingly.

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

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

18 If furthermore the Adaptation Sets are period-continuous, i.e. the presentation times are  
19 continuous and this is signalled in the MPD, then the client shall seamlessly play the con-  
20 tent across the Period boundary under the constraints in section 4.3.3.3.2. Most suitably the  
21 client may continue playing the Representation in the Adaptation Set with the same @id,  
22 but there is no guarantee that this Representation is available. In this case the client shall  
23 seamlessly switch to any other Representation in the Adaptation Set.

#### 24 **4.3.4.4. Joining, Initial Buffering and Playout Recommendations**

##### 25 **4.3.4.4.1. General**

26 A DASH client should start playout from:

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

##### 29 **4.3.4.4.2. Joining at the live edge**

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

- 31 • Every client participating in the service commits to the same presentation delay (PD) rela-  
32 tive to the announced segment availability start time at start-up and in continuous presen-  
33 tation, possible using one suggested by the Content Provider and then attempts to minimise  
34 start-up latency and maintain the buffer. The content provider may have provided the  
35 **MPD@suggestedPresentationDelay** (SPD) or may have provided this value by  
36 other means outside the DASH formats. The content author should be aware that the client  
37 may ignore the presence of **MPD@suggestedPresentationDelay** and may choose  
38 its own suitable playout scheduling.
- 39 • The client individually picks the presentation delay (PD) in order to maximize stable qual-  
40 ity and does this dependent on its access, user preferences and other considerations.

---

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

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

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

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

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

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

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

#### 26 **4.3.4.5. Requirements and Recommendations**

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

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

#### 31 **4.3.5. Additional DVB-DASH alignment aspects**

32 For alignment with DVB-DASH [43], the following should be considered:

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

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

---

### 1 **4.3.6. Considerations on live edge**

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

- 4 • If the MPD contains a `@publishTime` attribute with value `PUBT`, then at the publication  
5 of the MPD all Segments according to the computation in section 4.3.4.2 and 4.3.4.3 with  
6 `NOW` set to `PUBT` shall be available.
- 7 • If the MPD contains a `@publishTime` attribute with value `PUBT` and a Representation  
8 contains a Segment timeline with the `@r` attributed of the last **S** element being non-nega-  
9 tive, then the last Segment describe in this Segment timeline shall have a Segment availa-  
10 bility start time smaller than `PUBT` and the sum of the segment duration and the segment  
11 availability start time shall be larger than `PUBT`.

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

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

## 21 **4.4. Simple Live Service Offering including MPD Updates**

### 22 **4.4.1. Background and Assumptions**

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

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

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

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

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



1 According to the clustering in section 4.2, we distinguish two different types of live service offer-  
 2 ings:

- 3 • MPD controlled live service offering: In this case the DASH client typically frequently  
 4 polls the MPD update server whether an MPD update is available or the existing MPD can  
 5 still be used. The update frequency is controlled by MPD based on the attribute **MPD@min-**  
 6 **imumUpdatePeriod**. Such a service offering along with the client procedures is shown  
 7 in section 4.4.2.
- 8 • MPD and segment controlled offerings. In this case the DASH client needs to parse seg-  
 9 ments in order to identify MPD validity expirations and updates on the MPD update server.  
 10 MPD expiry events as described in section 5.10 of ISO/IEC 23009-1 "are pushed" to the  
 11 DASH client as parts of downloaded media segments. This offering along with the client  
 12 procedures is shown in section 4.5.

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

## 16 4.4.2. Preliminaries

### 17 4.4.2.1. MPD Information

18 As the MPD is typically updated over time on the server, the MPD that is accessed when joining  
 19 the service as well as the changes of the MPD are referred to as MPD instances in the following.

20 This expresses that for the same service, different MPDs exist depending on the time when the  
 21 service is consumed.

22 Assume that an MPD instance is present on the DASH server at a specific wall-clock time *NOW*.  
 23 For an MPD-based Live Service Offering, the MPD instance may among others contain infor-  
 24 mation as available in Table 9. Information included there may be used to compute a list of an-  
 25 nounced Segments, Segment Availability Times and URLs.

26 **Table 9 – Information related to Live Service Offering with MPD-controlled MPD Updates**

MPD Information	Status	Comment
<b>MPD@type</b>	mandatory, set to "dynamic"	the type of the Media Presenta- tion is dynamic, i.e. Seg- ments get available over time.
<b>MPD@availabilityStartTime</b>	mandatory	the start time is the anchor for the MPD in wall-clock time. The value is denoted as <i>AST</i> .
<b>MPD@minimumUpdatePeriod</b>	mandatory	this field is mandatory except for the case where the <b>MPD@mediaPresenta-</b> <b>tionDuration</b> is present. However, such an MPD falls then in an instance as docu- mented in section 4.3.

<b>Period</b> @start	mandatory	the start time of the Period relative to the MPD availability start time. The value is denoted as <i>PS</i> .
<b>SegmentTemplate</b> @media	mandatory	the template for the Media Segment
<b>SegmentTemplate</b> @startNumber	optional default	the number of the first segment in the Period. The value is denoted as <i>SSN</i> .
<b>SegmentTemplate</b> @duration	exactly one of <b>SegmentTemplate</b> @duration or <b>SegmentTemplate</b> . <b>SegmentTimeline</b> must be present	the duration of each Segment in units of a time. The value divided by the value of @timescale is denoted as <i>MD</i> [ <i>k</i> ] with <i>k</i> =1, 2, ... The segment timeline may contain some gaps.
<b>SegmentTemplate</b> . <b>SegmentTimeline</b>		

#### 1 4.4.2.2. Segment Information Derivation

2 Based on an MPD instance including information as documented in Table 9 and available at time  
3 *NOW* on the server, a DASH client may derive the information of the list of Segments for each  
4 Representation in each Period.

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

8 Note that with the MPD present on the server and *NOW* progressing, the Period end time is ex-  
9 tended. This issue is the only change compared to the segment information generation in section  
10 4.3.2.2.

#### 11 4.4.2.3. Some Special Cases

12 If the **MPD**@minimumUpdatePeriod is set to 0, then the MPD documents all available seg-  
13 ments on the server. In this case the @r count may be set accurately as the server knows all avail-  
14 able information.

### 15 4.4.3. Service Offering Requirements and Guidelines

#### 16 4.4.3.1. General

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

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

- 22 • start at wall-clock time *START*,
- 23 • location of the segments for each Representation at " [http://example.com/\\$Rep-](http://example.com/$RepresentationID/$Number$)  
24 [resentationID/\\$Number\\$](http://example.com/$RepresentationID/$Number$),"

1 a service provider may offer an MPD with values according to Table 10.

2 **Table 10 – Basic Service Offering with MPD Updates**

MPD Information	Value
<b>MPD</b> @type	dynamic
<b>MPD</b> @availabilityStartTime	START
<b>MPD</b> @publishTime	PUBTIME1
<b>MPD</b> @minimumUpdatePeriod	MUP
<b>MPD</b> .BaseURL	"http://example.com/"
<b>Period</b> @start	PSTART
<b>SegmentTemplate</b> @media	"\$RepresentationID\$/\$Number\$"
<b>SegmentTemplate</b> @startNumber	1
<b>SegmentTemplate</b> @duration	SDURATION

3 According to the work-flow shown in Annex B,

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

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

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

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

18 The MPD must be published latest at the end of the Media Presentation minus the value of *MUP*,  
19 i.e.  $PUBTIME2 \leq START + PSTART + PDURATION - MUP$ .

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

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

27 Track fragment decode times should not roll over and should not exceed  $2^{53}$  (due to observed  
28 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.

---

#### 1 4.4.3.4. Usage of Segment Timeline

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

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

13 An example for such an offering is shown in Table 11 where the RVALUE needs to be increased  
14 by 1 for each newly published segment.

15 **Table 11 – 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

#### 16 4.4.3.5. Last Segment Message

17 If the @segmentProfiles contains the 'lmsg' brand for a certain Representation, then the  
18 'lmsg' brand for signaling the last segment shall be applied for any content with MPD@mini-  
19 mumUpdatePeriod present and the MPD@type="dynamic".

20 DASH clients operating based on such an MPD and consuming the service at the live edge typi-  
21 cally need to request a new MPD prior to downloading a new segment. However, in order to min-  
22 imise MPD requests and resulting traffic load, the client may use one or more of the following  
23 optimisations:

- 24 • If the client fetches the MPD using HTTP, the client should use conditional GET methods  
25 as specified in RFC 7232 [24] to reduce unnecessary network usage in the downlink.
- 26 • If the @segmentProfiles contains the 'lmsg' brand clients may also rely on the  
27 'lmsg' message and request a new MPD only in case a segment is received with an  
28 'lmsg' brand. Otherwise the client may use template constructions to continue determin-  
29 ing the URL and the segment availability start time of segments.

---

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

#### 8 **4.4.4. MPD-based Live Client Operation based on MPD**

9 In an extension to the description in section 4.3.4.1 and section 4.3.4.3, the client now has access  
10 to an MPD and the MPD contains the **MPD@minimumUpdatePeriod**, for example following  
11 the parameters in Table 10. The start time of each Period is computed as period start time  $PSwc[i]$   
12 and the MPD-URL does not include any fragment parameters according to section 4.3.3.5.

13 The client fetches an MPD with parameters in Table 9 access to the MPD at time  $FetchTime$ , at  
14 its initial location if no **MPD.Location** element is present, or at a location specified in any pre-  
15 sent **MPD.Location** element.  $FetchTime$  is defined as the time at which the server processes  
16 the request for the MPD from the client. The client typically should not use the time at which it  
17 actually successfully received the MPD, but should take into account delay due to MPD delivery  
18 and processing. The fetch is considered successful either if the client obtains an updated MPD or  
19 the client verifies that the MPD has not been updated since the previous fetching.

20 If the client fetches the MPD using HTTP, the client should use conditional GET methods as spec-  
21 ified in RFC 7232 [24] to reduce unnecessary network usage in the downlink.

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

23 the client creates a list of accessible Segments at least for each selected Representation  
24 taking into account the information in the MPD as documented in Table 9 and the current  
25 time  $NOW$  by using the Period end time of the last Period as  $FetchTime + MUP$ .

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

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

33 If the client fetches the updated MPD using HTTP, the client should use conditional GET  
34 methods as specified in in RFC 7232 [24] to reduce unnecessary network usage in the  
35 downlink.

36 The client parses the MPD and generates a new segment list based on the new  $FetchTime$   
37 and  $MUP$  of the updated MPD. The client searches for the currently consumed Adaptation  
38 Sets and Representations and continues the process of downloading segments based on the  
39 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 12 – Service Offering with MPD and Segment-based Live Services

MPD Information	Status	Comment
<b>MPD@type</b>	mandatory, set to "dynamic"	the type of the Media Presentation is dynamic, i.e. Segments get available over time.
<b>MPD@publishTime</b>	mandatory	specifies the wall-clock time when the MPD was generated and published at the origin server. MPDs with a later value of <b>@publishTime</b> shall be an update as defined in 5.4 to MPDs with earlier <b>@publishTime</b> .
<b>MPD@availabilityStartTime</b>	mandatory	the start time is the anchor for the MPD in wall-clock time. The value is denoted as <i>AST</i> .

<b>MPD</b> @minimumUpdatePeriod	mandatory	recommended/mandate to be set to 0 to indicate that frequent DASH events may occur
<b>Period</b> @start	mandatory	the start time of the Period relative to the MPD availability start time. The value is denoted as <i>PS</i> .
<b>AdaptationSet.InbandEventStream</b>	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.
<b>SegmentTemplate</b> @media	mandatory	the template for the Media Segment
<b>SegmentTemplate</b> @startNumber	optional default	The number of the first segment in the Period. The value is denoted as <i>SSN</i> .
<b>SegmentTemplate</b> @duration	exactly one of <b>SegmentTemplate</b> @duration or <b>SegmentTemplate.SegmentTimeline</b> must be present	the duration of each Segment in units of a time. The value divided by the value of @timescale is denoted as <i>MD[k]</i> with k=1, 2, ... The segment timeline may contain some gaps.
<b>SegmentTemplate.SegmentTimeline</b>		

#### 1 4.5.1.2. Segment Information Derivation

2 Based on an MPD instance including information as documented in Table 9 and available at time  
3 *NOW* on the server, a DASH client may derive the information of the list of Segments for each  
4 Representation in each Period.

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

8 Note that with the MPD present on the server and *NOW* progressing, the Period end time is ex-  
9 tended. This issue is the only change compared to the segment information generation in section  
10 4.3.2.2.

11 If the **MPD**@minimumUpdatePeriod is set to 0, then the MPD documents all available seg-  
12 ments on the server. In this case the @r count may be set accurately as the server knows all avail-  
13 able information.



---

## 1 4.5.2. Service Offering Requirements and Guidelines

### 2 4.5.2.1. Background

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

6 DASH specific events that are of relevance for the DASH client are signalled in the MPD. The  
7 URN "urn:mpeg:dash:event:2012" is defined to identify the event scheme defined in Ta-  
8 ble 10.

9 **Table 13 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 <b>message_data</b> 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 <b>message_data</b> field.

10 Note: DVB DASH specification [43] does not include the value 3.

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

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

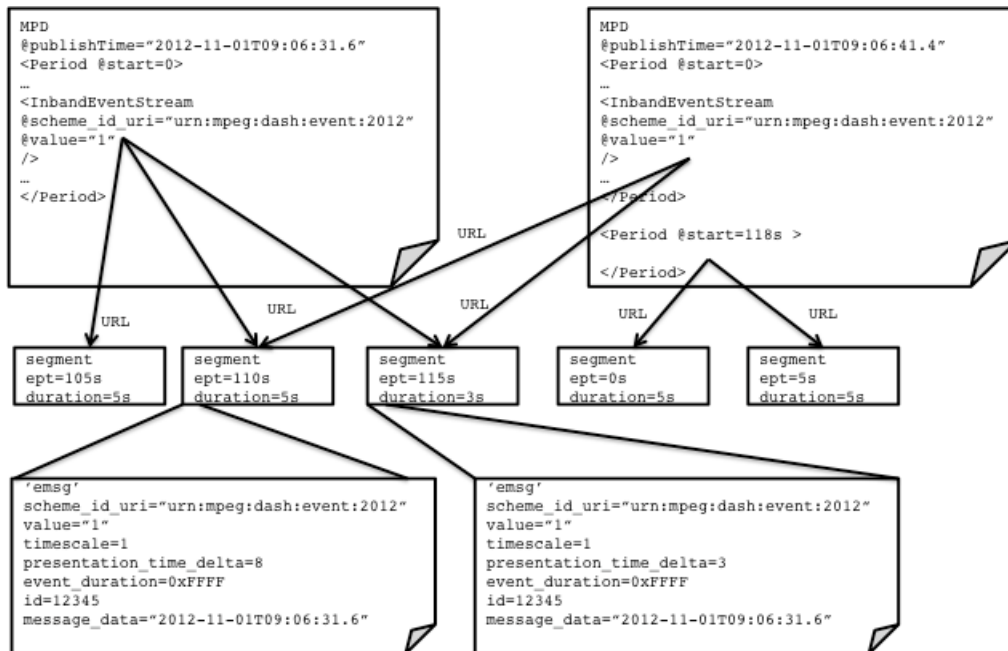


Figure 4 Example for MPD validity expiration to signal new Period

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.

1

Table 14 – Basic Service Offering with Inband Events

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

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

4 In this case the following shall apply:

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

16 the following shall be done

- 17 • a new MPD shall be published with a new publish time **MPD@publishTime**
- 18 • an 'emsg' box shall be added to each segment of each Representation that con-  
19 tains an **InbandEventStream** element with
  - 20 ○ `scheme_id_uri = "urn:mpeg:dash:event:2012"`
  - 21 ○ `@value` either set to 1 or set to 3
  - 22 ○ If `@value` set to 1 or 3
    - 23 ▪ the value of the **MPD@publishTime** of the previous MPD as the  
24 `message_data`

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

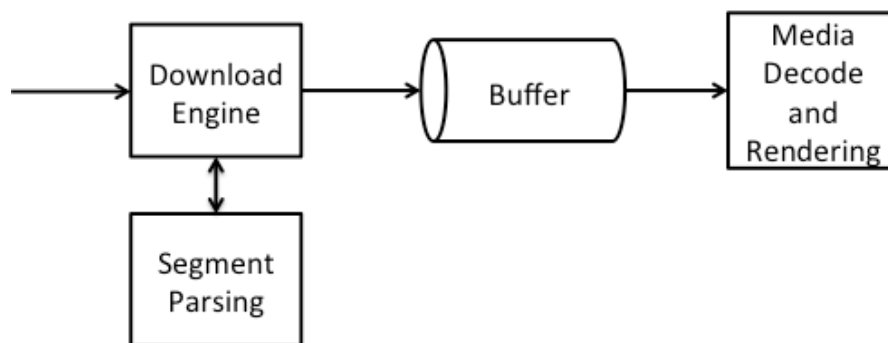
---

### 1 4.5.3. Client Requirements and Guidelines

#### 2 4.5.3.1. Introduction

3 A DASH client is guided by the information provided in the MPD. An advanced client model is  
4 shown in Figure 6. In contrast to the client in section 4.4.3.5, the advanced client requires parsing  
5 of segments in order to determine the following information:

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



11  
12 **Figure 6 Advanced Client Model**

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

- 15 • **MPD@minimumUpdatePeriod** is set to 0
- 16 • **MPD@publishTime** is included and the value is set to PUBTIME
- 17 • At least on Representation is present that contains an **InbandEventStream** element  
18 with scheme\_id\_uri="urn:mpeg:dash:event:2012" and @value either set  
19 to 1 or set to 3.
- 20 • Either the @duration or **SegmentTimeline** for the Representation is present.

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

#### 24 4.5.3.2. MPD Validity expiration and Updates

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

- 29 • ept the earliest presentation time of the media segment
- 30 • dur the media presentation duration of the media segment

---

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

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

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

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

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

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

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

23

24 NOTE: The DVB DASH profile [43] explicitly forbids downloading a Representation solely to gain access  
25 to an Inband Event Stream contained within it. For reference, the relevant part of the DVB DASH specifica-  
26 tion is section 9.1.6.

### 27 4.5.3.3. Extended Segment Information

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

- 31 • `ept` the earliest presentation time of the media segment
  - 32 ○ if the Segment Index is present use the Segments Index
  - 33 ○ if not use the `baseMediaDecodeTime` in 'tfdt' of the first movie frag-  
34 ment as the earliest presentation time
- 35 • `dur` the media presentation duration of the media segment
  - 36 ○ if the Segment Index is present use the Segments Index
  - 37 ○ if not use aggregated sample durations of the first movie fragment as the dura-  
38 tion

---

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

## 8 **4.6. Provisioning of Live Content in On-Demand Mode**

### 9 **4.6.1. Scenario**

10 A common scenario for DASH distribution results that a live generated service is also made avail-  
11 able for On-Demand offering after the live program is completed. The typical scenario is as fol-  
12 lows:

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

### 20 **4.6.2. Content Offering Requirements and Recommendations**

21 In order to provide live content as On-Demand in the above scenario, the following is recom-  
22 mended:

- 23 - The same Segments as generated for the live distribution are reused also for static distri-  
24 bution.
- 25 - Typically, the Segments also will have the same URL in order to exploit caching ad-  
26 vantages.
- 27 - An MPD should be generated latest at the end of the live session, but also may be created  
28 during an ongoing live session to document a certain window of the program that is offered  
29 for On-Demand.
- 30 - A new MPD is generated that should contain the following information
  - 31 o The **MPD@type** is set to *static*.
  - 32 o The **MPD@availabilityStartTime** may be set to any time in the past, for  
33 example the time of the original “live” MPD may be reused.
  - 34 o As profile, the simple live profile may be used
  - 35 o The attributes **@timeShiftBufferDepth** and **@minimumUpdatePeriod**  
36 are not present (in contrast to the live MPD) and a **@mediaPresentationDu-**  
37 **ration** attribute is added.
  - 38 o The window offered by the MPD is expressed by appropriately setting the **Pe-**  
39 **riod@start** value (including the presentation time offset and the start number)  
40 and the **@mediaPresentationDuration** attribute. The wall-clock time  
41 should be maintained by offsetting the **Period@start** without changing the  
42 **MPD@availabilityStartTime**.

- 
- 1       ○ Content may be offered in the same Period structure as for live or in a different  
2       one.
  - 3           ▪ If Periods are continuous, it is preferable to remove the Period structure.
  - 4           ▪ If new Periods are added for Ad Insertion, the Periods preferably be added  
5           in a way that they are at Segment boundaries.
  - 6       ○ Independent whether the @duration attribute or the **SegmentTimeline** ele-  
7       ment was used for the dynamic distribution, the static distribution version may  
8       have a **SegmentTimeline** with accurate timing to support seeking and to pos-  
9       sibly also signal any gaps in the Segment timeline. To obtain the accurate timeline,  
10       the segments may have to be parsed (at least up to the tfdt) to extract the duration  
11       of each Segment.
  - 12       ○ The same templating mode as used in the live service should also be used for static  
13       distribution.
  - 14       ○ DASH Event streams (i.e., MPD validity expirations) should not be present in the  
15       MPD.

### 16 **4.6.3. Client Behavior**

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

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

## 25 **4.7. Availability Time Synchronization between Client and Server**

### 26 **4.7.1. Background**

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

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

### 34 **4.7.2. Service Provider Requirements and Guidelines**

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

- 37       • The segment availability times announced in the MPD should be generated from a device  
38       that is synchronized to a globally accurate timing source, preferably using NTP.
- 39       • The MPD should contain at least one **UTCTiming** element with @schemeIdURI set to  
40       one of the following:
  - 41           ○ urn:mpeg:dash:utc:ntp:2014

- 
- 1       o urn:mpeg:dash:utc:http-head:2014
  - 2       o urn:mpeg:dash:utc:http-xdate:2014
  - 3       o urn:mpeg:dash:utc:http-iso:2014
  - 4       o urn:mpeg:dash:utc:http-ntp:2014
  - 5       • If the MPD does not contain any element **UTCTiming** then the segments shall be availa-
  - 6       ble latest at the announced segment availability time using a globally accurate timing
  - 7       source.
  - 8       • If the MPD contains an element **UTCTiming** then
  - 9       o the announced timing information in the **UTCTiming** shall be accessible to the
  - 10       DASH client, and
  - 11       o the segments shall be available latest at the announced segment availability time
  - 12       in the MPD for any device that uses one of announced time synchronization meth-
  - 13       ods at the same time.

14 If urn:mpeg:dash:utc:http-head:2014 is used, then the server specified in the @value  
15 attribute of the **UTCTiming** element should be the server hosting the DASH segments such that  
16 with each request the Date general-header field in the HTTP header (see in RFC 7231 [23], section  
17 7.1.12) can be used by the client to maintain synchronization.

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

21 A leap second is added to UTC every 18 months on average. A service provider should take into  
22 account the considerations in RFC 7164 [51].

23 The MPD time does not track leap seconds. If these occur during a live service they may advance  
24 or retard the media against the real time.

### 25 **4.7.3. Client Requirements and Guidelines**

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

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

36               Note: The DVB DASH [43] spec requires support for http-xdate and http-head but al-  
37               lows content providers to include others in addition, and allows clients to choose others in prefer-  
38               ence if they wish. For details, refer to section 4.7 of the DVB DASH specification.

- 39       • The client may take into account the accuracy of the timing source as well as any trans-  
40       mission 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 RFC 7231 [23]) 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 cases the client should check if the precision of the time synchronization to a globally accurate time standard 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.
- 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.

---

### 1 **4.8.3. Synchronization Loss of Segmenter**

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

- 4 1. The server is required to always offer a conforming media stream. In case the input  
5 stream or encoder is lost, the content author may always add dummy content. This may  
6 be done using a separate Period structure and is possible without any modifications of the  
7 standard.
- 8 2. Early Terminated Periods as included Cor.1 of the second edition of ISO/IEC 23009-1.  
9 Early Terminated Periods may be added that contain both **Period@start** and **Pe-**  
10 **riod@duration**. This expresses that for this Period no media is present at least for the  
11 time as expressed by the @duration attribute. Such Periods should only be used if Media  
12 Presentation author is experiencing issues in generating media, e.g. due to failures of a live  
13 feed. The MPD is updated using the @minimumUpdatePeriod, i.e. the timeline is  
14 progressing. This permits server to signal that there is an outage of media generation, but  
15 that the service is continuing. It is then up to the client to take appropriate actions.

### 16 **4.8.4. Encoder Clock Drift**

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

### 23 **4.8.5. Segment Unavailability**

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

### 29 **4.8.6. Swapping across Redundant Tools**

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

- 32 1. the content author is offering the service redundant to the client (for example using multi-  
33 ple Base URLs) and the client determines the availability of one or the other. This may be  
34 possible under certain circumstances
- 35 2. Periods may be inserted at a swap instance in order to provide the new information after  
36 swap. If possible, the offering may be continuous, but the offering may also be non-con-  
37 tinuous from a media time perspective.

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

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

#### 7        **4.8.7.        Service Provider Requirements and Guidelines**

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

#### 9        **4.8.8.        Client Requirements and Guidelines**

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

### 12       **4.9.       Interoperability Aspects**

#### 13       **4.9.1.       Introduction**

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

#### 16       **4.9.2.       Simple Live Operation**

##### 17       **4.9.2.1.       Definition**

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

##### 22       **4.9.2.2.       Service Requirements and Recommendations**

23       Service offerings conforming to this operation shall follow

- 24       • The general requirements and guidelines in section 4.3.3
- 25       • the MPD Update requirements and guidelines in section 4.4.3
- 26       • the requirements and guidelines for service offering of live content in on-demand mode in  
27       section 4.6.2
- 28       • the synchronization requirements and guidelines in section 4.7.2
- 29       • the robustness requirements and guidelines in section 4.8.7

##### 30       **4.9.2.3.       Client Requirements and Recommendations**

31       Clients claiming conformance to this operation shall follow

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

---

## 1 **4.9.3. Main Live Operation**

### 2 **4.9.3.1. Definition**

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

### 6 **4.9.3.2. Service Requirements and Recommendations**

7 Service offerings claiming conformance to main live shall follow

- 8 • the requirements and guidelines in section 4.3.3
- 9 • either
  - 10 ○ the requirements and guidelines in section 4.4.3. Note that in this case no profile
  - 11 identifier needs to be added.
- 12 • or
  - 13 ○ the segment-based MPD update requirements and guidelines in section 4.5.2. In
  - 14 this case the profile identifier shall be added.
- 15 • the requirements and guidelines for service offering of live content in on-demand mode in
- 16 section 4.6.2
- 17 • the synchronization requirements and guidelines in section 4.7.2
- 18 • the robustness requirements and guidelines in section 4.8.7

### 19 **4.9.3.3. Client Requirements and Recommendations**

20 Clients claiming conformance to main live shall follow

- 21 • the requirements and guidelines in section 4.3.4,
- 22 • the MPD-update requirements and guidelines in section 4.4.3.5,
- 23 • the segment-based MPD update requirements and guidelines in section 4.5.3,
- 24 • the requirements and guidelines for service offering of live content in on-demand mode in
- 25 section 4.6.3.
- 26 • the synchronization requirements and guidelines in section 4.7.3,
- 27 • the robustness requirements and guidelines in section 4.8.8.

## 28 **5. Ad Insertion in DASH**

### 29 **5.1. Introduction**

#### 30 **5.1.1. General**

31 This section provides recommendations for implementing ad insertion in DASH. Specifically, it  
32 defines the reference architecture and interoperability points for a DASH-based ad insertion solu-  
33 tion.

34 The baseline reference architecture addresses both server-based and app-based scenarios. The for-  
35 mer approach is what is typically used for Apple HLS, while the latter is typically used with Mi-  
36 crosoft SmoothStreaming and Adobe HDS.

---

## 1 5.1.2. Definitions

2 The following definitions are used in this section:

3 **Ad Break:** A location or point in time where one or more ads may be scheduled for delivery; same as  
4 *avail* and *placement opportunity*.

5 **Ad Decision Service:** functional entity that decides which ad(s) will be shown to the user. It  
6 interfaces deployment-specific and are out of scope for this document.

7 **Ad Management Module:** logical service that, given cue data, communicates with the ad decision  
8 service and determines which advertisement content (if at all) should be presented during the ad  
9 break described in the cue data.

10 **Cue:** indication of time and parameters of the upcoming ad break. Note that cues can indicate a  
11 pending switch to an ad break, pending switch to the next ad within an ad break, and pending switch  
12 from an ad break to the main content.

13 **CDN node:** functional entity returning a segment on request from DASH client. There are no  
14 assumptions on location of the node.

15 **Packager:** functional entity that processes conditioned content and produces media segments suitable  
16 for consumption by a DASH client. This entity is also known as fragmenter, encapsulator, or  
17 segmenter. Packager does not communicate directly with the origin server – its output is written to  
18 the origin server's storage.

19 **Origin:** functional entity that contains all media segments indicated in the MPD, and is the fallback if  
20 CDN nodes are unable to provide a cached version of the segment on client request.

21 **Splice Point:** point in media content where its stream may be switched to the stream of another  
22 content, e.g. to an ad.

23 **MPD Generator:** functional entity returning an MPD on request from DASH client. It may be  
24 generating an MPD on the fly or returning a cached one.

25 **XLink resolver:** functional entity which returns one or more remote elements on request from  
26 DASH client.

## 27 5.1.3. DASH Concepts

### 28 5.1.3.1. Introduction

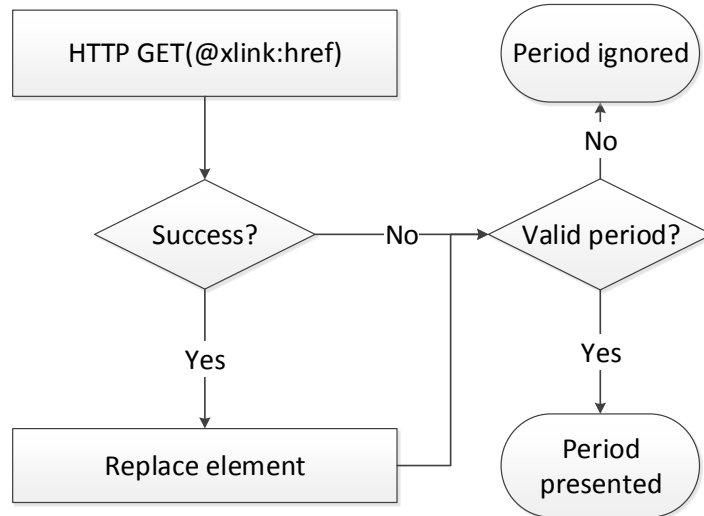
29 DASH ad insertion relies on several DASH tools defined in the second edition of ISO/IEC 23009-  
30 1 [5], which are introduced in this section. The correspondence between these tools and ad inser-  
31 tion concepts are explained below.

### 32 5.1.3.2. Remote Elements

33 *Remote elements* are elements that are not fully contained in the MPD document but are referenced  
34 in the MPD with an HTTP-URL using a simplified profile of XLink.

35 A remote element has two attributes, `@xlink:href` and `@xlink:actuate`. `@xlink:href`  
36 contains the URL for the complete element, while `@xlink:actuate` specifies the resolution  
37 model. The value "onLoad" requires immediate resolution at MPD parse time, while "onRe-  
38 quest" allows deferred resolution at a time when an XML parser accesses the remote element.  
39 In this text we assume deferred resolution of remote elements, unless explicitly stated otherwise.  
40 While there is no explicit timing model for earliest time when deferred resolution can occur, the  
41 specification strongly suggests it should be close to the expected playout time of the corresponding

1 Period. A reasonable approach is to choose the resolution at the nominal download time of the  
 2 Segment.



3  
 4 **Figure 7: XLink resolution**

5 Resolution (a.k.a. dereferencing) consists of two steps. Firstly, a DASH client issues an HTTP  
 6 GET request to the URL contained in the @xlink:href, attribute of the in-MPD element, and  
 7 the XLink resolver responds with a remote element entity in the response content. In case of error  
 8 response or syntactically invalid remote element entity, the @xlink:href and  
 9 @xlink:actuate attributes the client shall remove the in-MPD element.

10 If the value of the @xlink:href attribute is urn:mpeg:dash:resolve-to-zero:2013,  
 11 HTTP GET request is not issued, and the in-MPD element shall be removed from the MPD. This  
 12 special case is used when a remote element can be accessed (and resolved) only once during the  
 13 time at which a given version of MPD is valid.

14 If a syntactically valid remote element entity was received, the DASH client will replace in-MPD  
 15 element with remote period entity.

16 Once a remote element entity is resolved into a fully specified element, it may contain an  
 17 @xlink:href attribute with @xlink:actuate set to 'onRequest', which contains a new  
 18 XLink URL allowing repeated resolution.

19 Note that the only information passed from the DASH client to the XLink resolver is encoded  
 20 within the URL. Hence there may be a need to incorporate parameters into it, such as splice time  
 21 (i.e., PeriodStart for the remote period) or cue message.

22 **5.1.3.3. Periods**

23 **5.1.3.3.1. Timing**

24 Periods are time-delimited parts of a DASH Media Presentation. The value of PeriodStart can be  
 25 explicitly stated using the Period@start attribute or indirectly computed using Period@du-  
 26 ration of the previous Periods.

27 Precise period duration of period  $i$  is given by  $PeriodStart(i+1) - PeriodStart(i)$ . This can accom-  
 28 modate the case where media duration of period  $i$  is slightly longer than the period itself, in which  
 29 case a client will schedule the start of media presentation for period  $i+1$  at time  $PeriodStart(i+1)$ .

30 **Representation@presentationTimeOffset** specifies the value of the presentation  
 31 time at  $PeriodStart(i)$ .

---

#### 1 5.1.3.3.2. Segment Availability

2 In case of dynamic MPDs, Period-level **BaseURL**@availabilityTimeOffset allow earlier  
3 availability start times. A shorthand notation @availabilityTimeOffset="INF" at a Pe-  
4 riod-level **BaseURL** indicates that the segments within this period are available at least as long as  
5 the current MPD is valid. This is the case with stored ad content. Note that DASH also allows  
6 specification of @availabilityTimeOffset at Adaptation Set and Representation level.

#### 7 5.1.3.3.3. Seamless transition

8 The DASH specification says nothing about Period transitions – i.e., there are no guarantees for  
9 seamless continuation of playout across the period boundaries. Content conditioning and receiver  
10 capability requirements should be defined for applications relying on this functionality. However,  
11 Period continuity may be used and signaled using the tools defined in ISO/IEC 23009-  
12 1:2014/Amd.3 [5].

#### 13 5.1.3.3.4. Period labeling

14 Period-level **AssetIdentifier** descriptors identify the asset to which a given Period belongs.  
15 Beyond identification, this can be used for implementation of client functionality that depends on  
16 distinguishing between ads and main content (e.g. progress bar and random access).

#### 17 5.1.3.4. DASH events

18 DASH events are messages having type, timing and optional payload. They can appear either in  
19 MPD (as period-level event stream) or inband, as ISO-BMFF boxes of type `emsg`. The `emsg`  
20 boxes shall be placed at the very beginning of the Segment, i.e. prior to any media data, so that  
21 DASH client needs a minimal amount of parsing to detect them.

22 DASH defines three events that are processed directly by a DASH client: MPD Validity Expira-  
23 tion, MPD Patch and MPD Update. All signal to the client that the MPD needs to be updated – by  
24 providing the publish time of the MPD that should be used, by providing an XML patch that can  
25 be applied to the client's in-memory representation of MPD, or by providing a complete new MPD.  
26 For details please see section 4.5.

27 User-defined events are also possible. The DASH client does not deal with them directly – they  
28 are passed to an application, or discarded if there is no application willing or registered to process  
29 these events. A possible client API would allow an application to register callbacks for specific  
30 event types. Such callback will be triggered when the DASH client parses the `emsg` box in a  
31 Segment, or when it parses the **Event** element in the MPD.

32 In the ad insertion context, user-defined events can be used to signal information, such as cue  
33 messages (e.g. SCTE 35 [55])

#### 34 5.1.3.5. MPD Updates

35 If **MPD**@minimumUpdatePeriod is present, the MPD can be periodically updated. These up-  
36 dates can be *synchronous*, in which case their frequency is limited by **MPD**@minimumUpdate-  
37 Period. In case of the main live profiles MPD updates may be triggered by DASH events. Fir  
38 details refer to section 4.5.

39 When new period containing stored ads is inserted into a linear program, and there is a need to  
40 unexpectedly alter this period the inserted media will not carry the `emsg` boxes – these will need  
41 to be inserted on-the-fly by proxies. In this case use of synchronous MPD updates may prove  
42 simpler.

---

1 **MPD**@publishTime provides versioning functionality: MPD with later publication times in-  
2 clude all information that was included all MPDs with earlier publication times.

### 3 **5.1.3.6. Session information**

4 In order to allow fine-grain targeting and personalization, the identity of the client/viewer, should  
5 be known i.e. maintain a notion of a session.

6 HTTP is a stateless protocol, however state can be preserved by the client and communicated to  
7 the server.

8 The simplest way of achieving this is use of cookies. According to RFC 6265 [42], cookies set via  
9 2xx, 4xx, and 5xx responses must be processed and have explicit timing and security model.

### 10 **5.1.3.7. Tracking and reporting**

11 The simplest tracking mechanism is server-side logging of HTTP GET requests. Knowing request  
12 times and correspondence of segment names to content constitutes an indication that a certain part  
13 of the content was requested. If MPDs (or remote element entities) are generated on the fly and  
14 identity of the requester is known, it is possible to provide more precise logging. Unfortunately  
15 this is a non-trivial operation, as same user may be requesting parts of content from different CDN  
16 nodes (or even different CDNs), hence log aggregation and processing will be needed.

17 Another approach is communicating with existing tracking server infrastructure using existing ex-  
18 ternal standards. An IAB VAST-based implementation is shown in section 5.3.3.7.

19 DASH Callback events are defined in ISO/IEC 23009-1:2014 AMD3 [5], are a simple native im-  
20 plementation of time-based impression reporting (e.g., quartiles). A callback event is a promise by  
21 the DASH client to issue an HTTP GET request to a provided URL at a given offset from *Period-*  
22 *Start*. The body of HTTP response is ignored.

## 23 **5.2. Architectures**

24 The possible architectures can be classified based on the location of component that communicates  
25 with the ad decision service: a *server-based* approach assumes a generic DASH client and all com-  
26 munication with ad decision services done at the server side (even if this communication is trig-  
27 gered by a client request for a segment, remote element, or an MPD. The *app-based* approach  
28 assumes an application running on the end device and controlling one or more generic DASH  
29 clients.

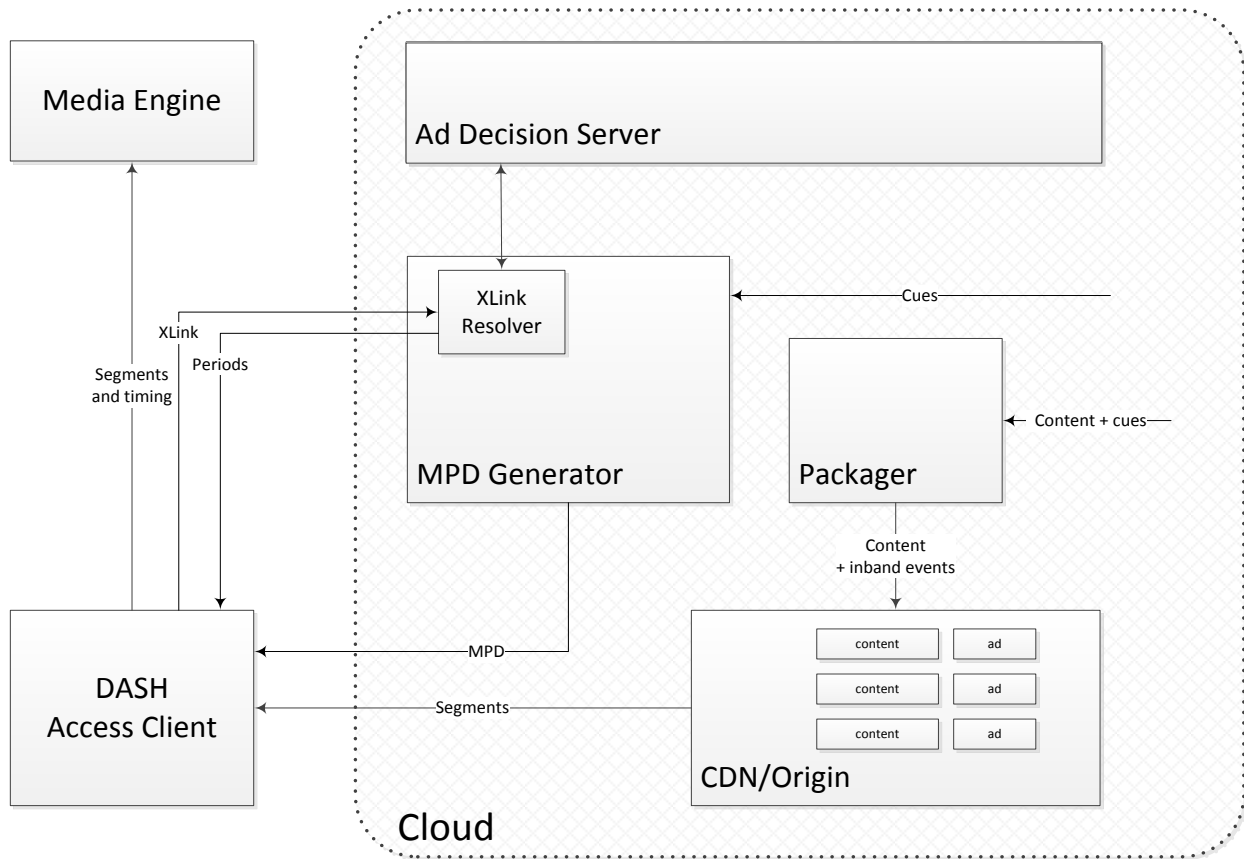
30 Yet another classification dimension is amount of media engines needed for a presentation – i.e.,  
31 whether parallel decoding needs to be done to allow seamless transition between the main and the  
32 inserted content, or content is conditioned well enough to make such transition possible with a  
33 single decoder.

34 Workflows can be roughly classified into *linear* and *elastic*. Linear workflows (e.g., live feed from  
35 an event) has ad breaks of known durations which have to be taken: main content will only resume  
36 after the end of the break and the programmer / operator needs to fill them with some inserted  
37 content. Elastic workflows assume that the duration of an ad break at a given cue location not  
38 fixed, thus the effective break length can vary (and can be zero if a break is not taken).



## 1 5.3. Server-based Architecture

### 2 5.3.1. Introduction



3  
4 **Figure 8: Server-based architecture**

5 In the server-based model, all ad-related information is expressed via MPD and segments, and ad  
6 decisions are triggered by client requests for MPDs and for resources described in them (Segments,  
7 remote periods).

8 The server-based model is inherently MPD-centric – all data needed to trigger ad decision is con-  
9 centrated in the MPD. In case where ad break location (i.e., its start time) is unknown at the MPD  
10 generation time, it is necessary to rely on MPD update functionality. The two possible ways of  
11 achieving these are described in 5.1.3.5.

12 In the live case, packager receives feed containing inband cues, such as MPEG-2 TS with SCTE  
13 35 cue messages [55]. The packager ingests content segments into the CDN. In the on demand  
14 case, cues can be provided out of band.

15 Ad management is located at the server side (i.e., in the cloud), thus all manifest and content con-  
16 ditioning is done at the server side.

### 17 5.3.2. Mapping to DASH

#### 18 5.3.2.1. Period elements

##### 19 5.3.2.1.1. General

20 A single ad is expressed as a single **Period** element.

---

1 Periods with content that is expected to be interrupted as a result of ad insertion should contain  
2 explicit start times (**Period**@start), rather than durations. This allows insertion of new periods  
3 without modifying the existing periods. If a period has media duration longer than the distance  
4 between the start of this period and the start of next period, use of start times implies that a client  
5 will start the playout of the next period at the time stated in the MPD, rather than after finishing  
6 the playout of the last segment.

7 An upcoming ad break is expressed as Period element(s), possibly remote.

#### 8 **5.3.2.1.2. Remote Period elements.**

9 Remote Periods are resolved on demand into one or more than one Period elements. It is possible  
10 to embed parameters from the cue message into the XLink URL of the corresponding remote pe-  
11 riod, in order to have them passed to the ad decision system via XLink resolver at resolution time.  
12 In an elastic workflow, when an ad break is not taken, the remote period will be resolved into a  
13 period with zero duration. This period element will contain no adaptation sets.

14 If a just-in-time remote Period dereferencing is required by use of @xlink:actuate="onRe-  
15 quest", MPD update containing a remote period should be triggered close enough to the intended  
16 splice time. This can be achieved using MPD Validity events and full-fledged MPD update, or  
17 using MPD Patch and MPD Update events (see sec. 5.1.3.5 and 5.1.3.4). However, due to security  
18 reasons MPD Patch and MPD Update events should only be used with great care.

19 In case of **Period**@xlink:actuate="onRequest", MPD update and XLink resolution  
20 should be done sufficiently early to ensure that there are no artefacts due to insufficient time given  
21 to download the inserted content. Care needs to be taken so that the client is given a sufficient  
22 amount of time to (a) request and receive MPD update, and (b) dereference the upcoming remote  
23 period.

24 NOTE: It may be operationally simpler to avoid use of **Period**@xlink:actu-  
25 ate="onRequest", dereferencing in case of live content.  
26

#### 27 **5.3.2.2. Asset Identifiers**

28 **AssetIdentifier** descriptors identify the asset to which a Period belongs. This can be used  
29 for implementation of client functionality that depends on distinguishing between ads and main  
30 content (e.g. progress bar).

31 Periods with same **AssetIdentifier** should have identical Adaptation Sets, Initialization  
32 Segments and same DRM information (i.e., DRM systems, licenses). This allows reuse of at least  
33 some initialization data across periods of the same asset, and ensures seamless continuation of  
34 playback if inserted periods have zero duration. Period continuity may be signaled.

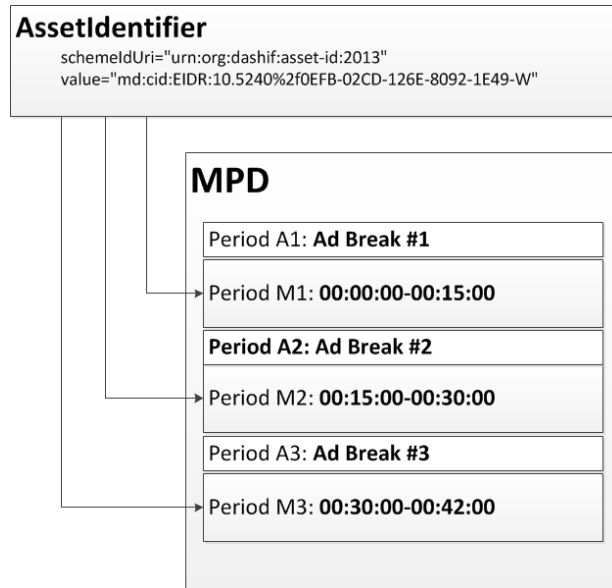


Figure 9 Using an Asset Identifier

1  
2

3

4 **5.3.2.3. MPD updates**

5 MPD updates are used to implement dynamic behavior. An updated MPD may have additional  
6 (possibly – remote) periods. Hence, MPD update should be triggered by the arrival of the first cue  
7 message for an upcoming ad break. Ad breaks can also be canceled prior to their start, and such  
8 cancellation will also trigger an MPD update.

9 Frequent regular MPD updates are sufficient for implementing dynamic ad insertion. Unfortun-  
10 ately they create an overhead of unnecessary MPD traffic – ad breaks are rare events, while MPD  
11 updates need to be frequent enough if a cue message is expected to arrive only several seconds  
12 before the splice point. Use of HTTP conditional GET requests (i.e., allowing the server to respond  
13 with "304 Not Modified" if MPD is unchanged) is helpful in reducing this overhead, but asynchro-  
14 nous MPD updates avoid this overhead entirely.

15 DASH events with scheme "urn:mpeg:dash:event:2013" are used to trigger asynchro-  
16 nous MPD updates.

17 The simple mapping of live inband cues in live content into DASH events is translating a single  
18 cue into an MPD Validity expiration event (which will cause an MPD update prior to the splice  
19 time). MPD Validity expiration events need to be sent early enough to allow the client request a  
20 new MPD, resolve XLink (which may entail communication between the resolver and ADS), and,  
21 finally, download the first segment of the upcoming ad in time to prevent disruption of service at  
22 the splice point.

23 If several `emsg` boxes are present in a segment and one of them is the MPD Validity Expiration  
24 event, `emsg` carrying it shall always appear first.

25 **5.3.2.4. MPD events**

26 In addition to tracking events (ad starts, quartile tracking, etc.) the server may also need to signal  
27 additional metadata to the video application. For example, an ad unit may contain not only inline  
28 linear ad content (that is to be played before, during, or after the main presentation), it may also  
29 contain a companion display ad that is to be shown at the same time as the video ad. It is important

1 that the server be able to signal both the presence of the companion ad and the additional tracking  
 2 and click-through metadata associated with the companion.  
 3 With that said, there is no need to have a generic DASH client implement this functionality – it is  
 4 enough to provide opaque information that the client would pass to an external module. Event  
 5 @schemeIdUri provides us with such addressing functionality, while MPD events allow us to  
 6 put opaque payloads into the MPD.

7 **5.3.3. Workflows**

8 **5.3.3.1. General**

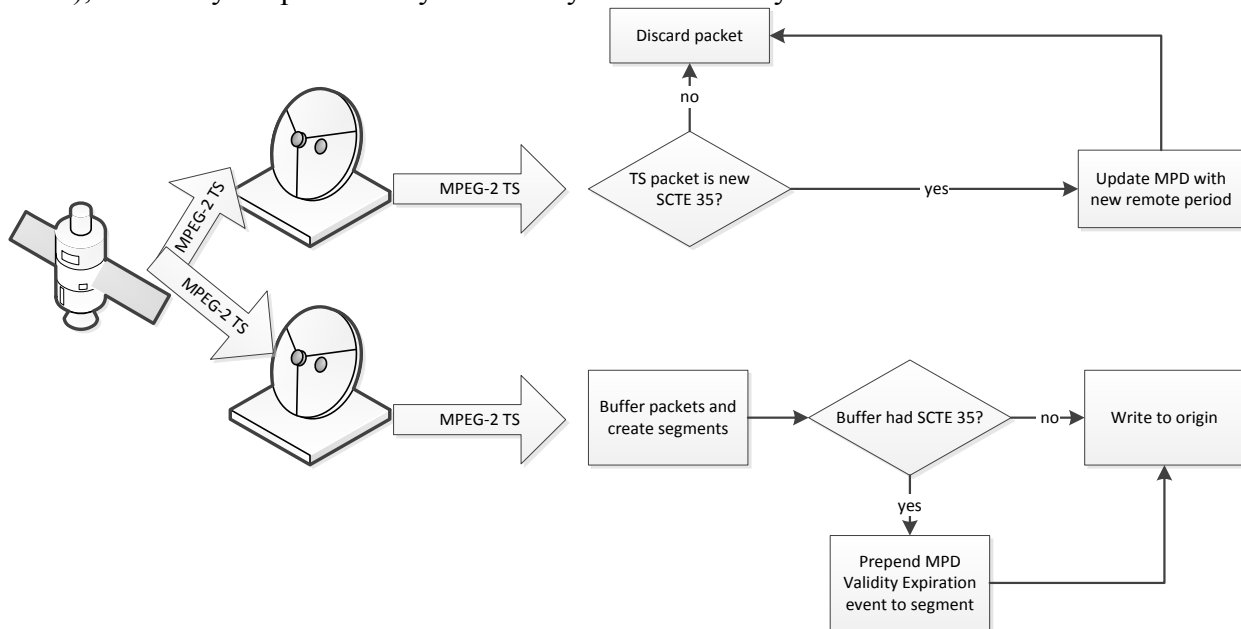
9 In the workflows below we assume that our inputs are MPEG-2 transport streams with embedded  
 10 SCTE 35 cue messages [55]. In our opinion this will be a frequently encountered deployment,  
 11 however any other in-band or out-of-band method of getting cue messages and any other input  
 12 format lend themselves into the same model.

13 **5.3.3.2. Linear**

14 A real-time MPEG-2 TS feed arrives at both packager and MPD generator. While real-time mul-  
 15 ticast feeds are a very frequently encountered case, the same workflow can apply to cases such as  
 16 ad replacement in a pre-recorded content (e.g., in time-shifting or PVR scenarios).

17 MPD generator generates dynamic MPDs. Packager creates DASH segments out of the arriving  
 18 feed and writes them into the origin server. Client periodically requests the MPDs so that it has  
 19 enough time to transition seamlessly into the ad period.

20 Packager and MPD generator may be tightly coupled (e.g. co-located on the same physical ma-  
 21 chine), or loosely coupled as they both are synchronized only to the clock of the feed.



22  
 23  
 24 **Figure 10: Live Workflow**

25 **5.3.3.2.1. Cue Interpretation by the MPD generator**

26 When an SCTE 35 cue message indicating an upcoming splice point is encountered by the MPD  
 27 generator, the latter creates a new MPD for the same program, adding a remote period to it.

---

1 The **Period**@start attribute of the inserted period has splice\_time() translated into the  
2 presentation timeline. Parameters derived from the cue message are inserted into the **Pe-**  
3 **riod**@xlink:href attribute of the inserted period. Examples below show architectures that  
4 allow finer targeting.

#### 5 **5.3.3.2.1.1. Example 1: Immediate ad decision**

6 MPD generator keeps an up-to-date template of an MPD. At each cue message arrival, the gener-  
7 ator updates its template. At each MPD request, the generator customizes the request based on the  
8 information known to it about the requesting client. The generator contacts ad decision server and  
9 produces one or more non-remote ad periods. In this case XLink is not needed.

#### 10 **5.3.3.2.1.2. Example 2: Stateful cue translation**

11 MPD generator keeps an up-to-date template of an MPD. At each cue message arrival, the gener-  
12 ator updates its template. At each MPD request, the generator customizes the request based on the  
13 information known to it about the requesting client.

14 The operator targets separately male and female audiences. Hence, the generator derives this from  
15 the information it has regarding the requesting client (see 5.1.3.6), and inserts an XLink URL with  
16 the query parameter ?gender=male for male viewers, and ?gender=female for the female  
17 viewers.

18 Note that this example also showcases poor privacy practices – would such approach be imple-  
19 mented, both parameter name and value should be encrypted or TLS-based communication should  
20 be used

#### 21 **5.3.3.2.1.3. Example 3: Stateless cue translation**

22 At cue message arrival, the MPD generator extracts the entire SCTE 35 splice\_info\_sec-  
23 tion (starting at the table\_id and ending with the CRC\_32) into a buffer. The buffer is then  
24 encoded into URL-safe base64url format according to RFC 4648 [61], and inserted into the XLink  
25 URL of a new remote Period element. splice\_time is translated into **Period**@start attrib-  
26 ute. The new MPD is pushed to the origin.

27 Note: this example is a straightforward port of the technique defined for SCTE 67 [56], but uses base64url  
28 and not base64 encoding as the section is included in a URI.

#### 29 **5.3.3.2.2. Cue Interpretation by the packager**

30 Cue interpretation by the packager is optional and is an optimization, rather than core functionality.  
31 On reception of an SCTE 35 cue message signaling an upcoming splice, an `emsg` with MPD  
32 Validity Expiration event is inserted into the first available segment. This event triggers an MPD  
33 update, and not an ad decision, hence the sum of the earliest presentation time of the `emsg`-  
34 bearing segment and the `emsg`.presentation\_time\_delta should be sufficiently ear-  
35 lier than the splice time. This provides the client with sufficient time to both fetch the MPD and  
36 resolve XLink.

37 splice\_time() of the cue message is translated into the media timeline, and last segment be-  
38 fore the splice point is identified. If needed, the packager can also finish the segment at the splice  
39 point and thus having a segment shorter than its target duration.

#### 40 **5.3.3.2.3. Multiple cue messages**

41 There is a practice of sending several SCTE 35 cue messages for the same splice point (e.g., the  
42 first message announces a splice in 6 seconds, the second arrives 2 seconds later and warns about

1 the same splice in 4 seconds, etc.). Both the packager and the MPD generator react on the same  
2 first message (the 6-sec warning in the example above), and do nothing about the following mes-  
3 sages.

#### 4 5.3.3.2.4. Cancellation

5 It is possible that the upcoming (and announced) insertion will be canceled (e.g., ad break needed  
6 to be postponed due to overtime). Cancellation is announced in a SCTE 35 cue message.

7 When cancellation is announced, the packager will insert the corresponding `emsg` event and the  
8 MPD generator will create a newer version of the MPD that does not contain the inserted period  
9 or sets its duration to zero. This implementation maintains a simpler less-coupled server side sys-  
10 tem at the price of an increase in traffic.

#### 11 5.3.3.2.5. Early termination

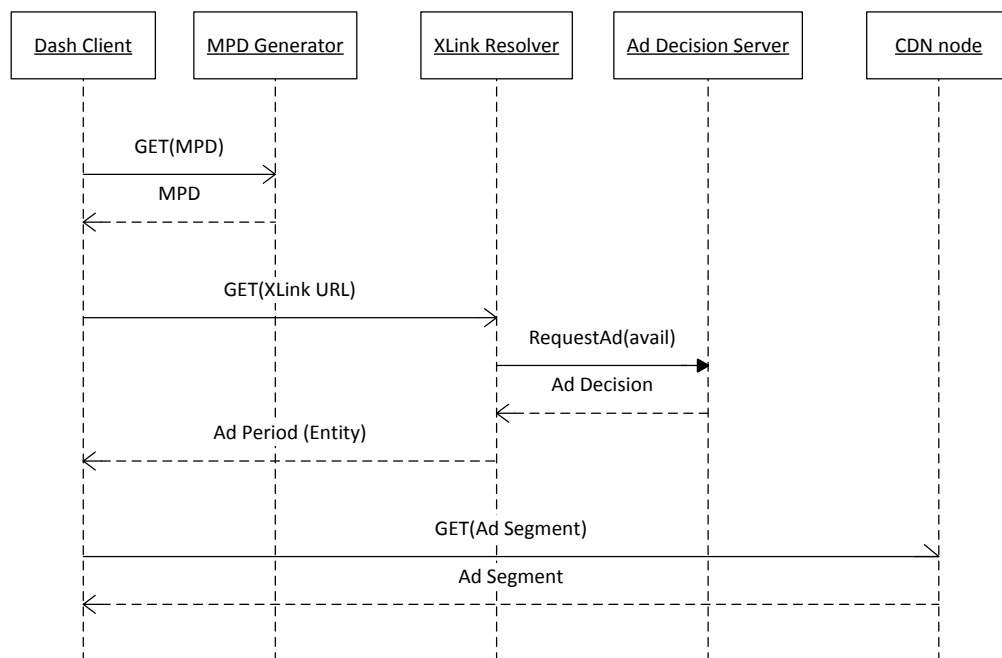
12 It is also possible that a planned ad break will need to be cut short – e.g., an ad will be cut short  
13 and there will be a switch to breaking news. The DASH translation of this would be creating an  
14 `emsg` at the packager and updating the MPD appropriately. Treatment of early termination here  
15 would be same as treatment of a switch from main content to an ad break.

16 It is easier to manipulate durations when **Period**@duration is absent and only **Pe-**  
17 **riod**@start is used – this way attributes already known to the DASH client don't change.

#### 18 5.3.3.2.6. Informational cue messages

19 SCTE 35 can be used for purposes unrelated to signaling of placement opportunities. Examples  
20 of such use are content identification and time-of-day signaling. Triggering MPD validity expira-  
21 tion and possibly XLink resolution in this case may be an overreaction.

#### 22 5.3.3.2.7. Ad decision



---

### Figure 11: Ad Decision

1  
2 A client will attempt to dereference a remote period element by issuing an HTTP GET for the URL  
3 that appears in **Period@xlink:href**. The HTTP server responding to this request (XLink re-  
4 solver) will contact the ad decision service, possibly passing it parameters known from the request  
5 URL and from client information available to it from the connection context. In case described in  
6 5.3.3.2.1.3, the XLink resolver has access to a complete SCTE 35 message that triggered the splice.  
7 The ad decision service response identifies the content that needs to be presented, and given this  
8 information the XLink resolver can generate one or more Period elements that would be then re-  
9 turned to the requesting DASH client.

10 A possible optimization is that resolved periods are cached – e.g. in case of 5.3.3.2.1.1 "male" and  
11 "female" versions of the content are only generated once in  $T$  seconds, with HTTP caching used  
12 to expire the cached periods after  $T$  seconds.

#### 13 5.3.3.3. On Demand

14 In a VoD scenario, cue locations are known ahead of time. They may be available multiplexed into  
15 the mezzanine file as SCTE 35 or SCTE 104, or may be provided via an out-of-band EDL.

16 In VoD workflows both cue locations and break durations are known, hence there is no need for a  
17 dynamic MPD. Thus cue interpretation (which is same as in 5.3.3.2) can occur only once and result  
18 in a static MPD that contains all remote elements with all Period elements having **Pe-**  
19 **riod@start** attribute present in the MPD.

20 In elastic workflows ad durations are unknown, thus despite our knowledge of cue locations within  
21 the main content it is impossible to build a complete presentation timeline. **Period@duration**  
22 needs to be used. Remote periods should be dereferenced only when needed for playout. In case  
23 of a “jump” – random access into an arbitrary point in the asset – it is a better practice not to  
24 dereference Period elements when it is possible to determine the period from which the playout  
25 starts using **Period@duration** and asset identifiers. The functionality described in 5.3.3.2 is  
26 sufficient to address on-demand cases, with the only difference that a client should be able to  
27 handle zero-duration periods that are a result of avails that are not taken.

#### 28 5.3.3.4. Capture to VoD

29 Capture to VoD use case is a hybrid between pure linear and on demand scenarios: linear content  
30 is recorded as it is broadcast, and is then accessible on demand. A typical requirement is to have  
31 the content available with the original ad for some time, after which ads can be replaced

32 There are two possible ways of implementing the capture-to-VoD workflow.

33 The simplest is treating capture-to-VoD content as plain VoD, and having the replacement policy  
34 implemented on the XLink resolver side. This way the same Period element(s) will be always  
35 returned to the same requester within the window where ad replacement is disallowed; while after  
36 this window the behavior will be same as for any on-demand content. An alternative implementa-  
37 tion is described in 5.3.3.5 below.

#### 38 5.3.3.5. Slates and ad replacement

39 A content provider (e.g., OTT) provides content with ad breaks filled with its own ads. An ISP is  
40 allowed to replace some of these with their own ads. Conceptually there is content with slates in  
41 place of ads, but all slates can be shown and only some can be replaced.

1 An ad break with a slate can be implemented as a valid in-MPD Period element that also has XLink  
2 attributes. If a slate is replaceable, XLink resolution will result in new Period element(s), if not –  
3 the slate is played out.

#### 4 **5.3.3.6. Blackouts and Alternative content**

5 In many cases broadcast content cannot be shown to a part of the audience due to contractual  
6 limitations (e.g., viewers located close to an MLB game will not be allowed to watch it, and will  
7 be shown some alternative content). While unrelated to ad insertion per se, this use case can be  
8 solved using the same “default content” approach, where the in-MPD content is the game and the  
9 alternative content will be returned by the XLink resolver if the latter determines (in some unspec-  
10 ified way) that the requester is in the blackout zone.

#### 11 **5.3.3.7. Tracking and reporting**

12 A Period, either local or a remote entity, may contain an EventStream element with an event con-  
13 taining IAB VAST 3.0 Ad element [54]. DASH client does not need to parse the information and  
14 act accordingly – if there is a listener to events of this type, this listener can use the VAST 3.0 Ad  
15 element to implement reporting, tracking and companion ads. The processing done by this listener  
16 does not have any influence on the DASH client, and same content would be presented to both  
17 “vanilla” DASH client and the player in which a VAST module registers with a DASH client a  
18 listener to the VAST 3.0 events.

19 An alternative implementation uses DASH Callback events. While DASH specification permits  
20 both inband and MPD Callback events, inband callback events shall not be used.

### 21 **5.3.4. Examples**

#### 22 **5.3.4.1. MPD with mid-roll ad breaks and default content**

23 In this example, a movie (“Top Gun”) is shown on a linear channel and has two mid-roll ad breaks.  
24 Both breaks have default content that will be played if the XLink resolver chooses not to return  
25 new Period element(s) or fails.

26 In case of the first ad break, SCTE 35 cue message is passed completely to the XLink resolver,  
27 together with the corresponding presentation time.

28 In case of the second ad break, proprietary parameters u and z describe the main content and the  
29 publishing site.

30

```
<?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=DAIAAAAAAAAAAAAAQAAZ_I0VniQAQAgBDVUVJQAAAAAH+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 start="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 12: Example of MPD for "Top Gun" movie

## 5.4. App-based Architecture

### 5.4.1. Introduction

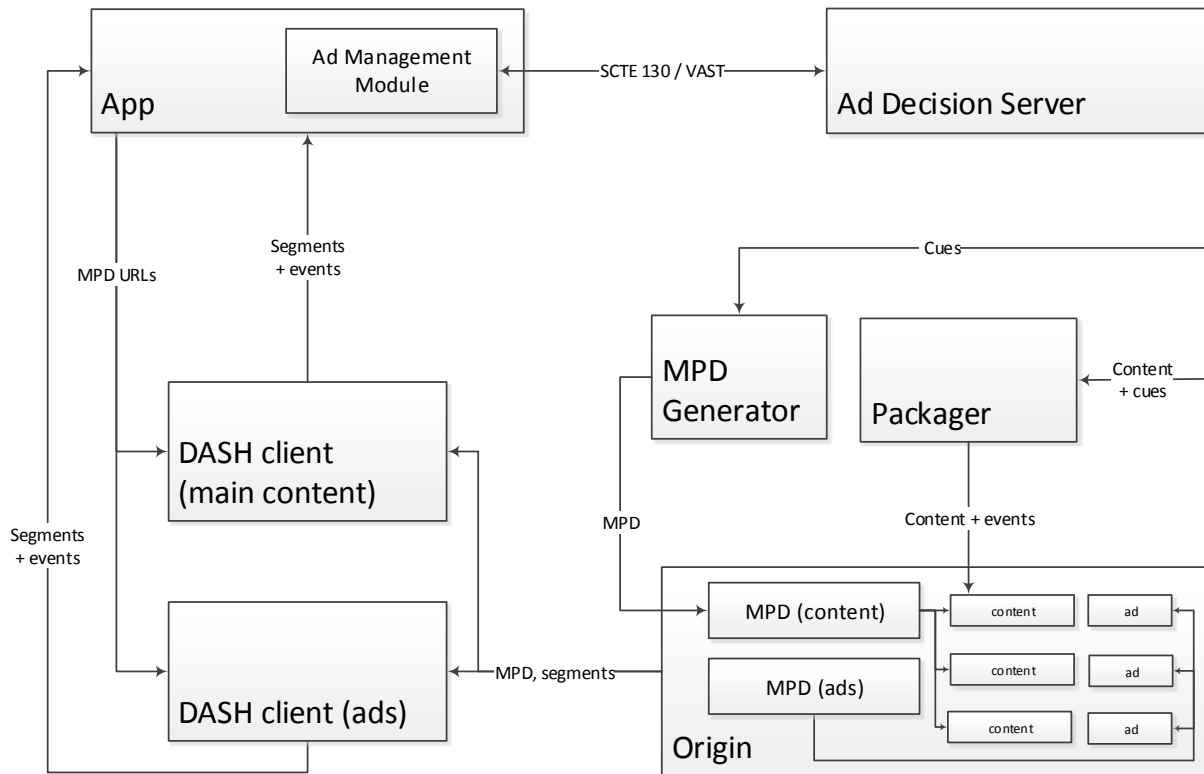


Figure 13: App-based architecture

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.

## 1 5.4.2. Mapping to DASH

2 This section details mapping of elements of the reference architecture into DASH concepts per the  
3 2<sup>nd</sup> edition of the specification (i.e., ISO/IEC 23009-1:2014).

### 4 5.4.2.1. MPD

5 Each ad decision results in a separate MPD. A single MPD contains either main content or inserted  
6 content; existence of multiple periods or/and remote periods is possible but not essential.

### 7 5.4.2.2. SCTE 35 events

#### 8 5.4.2.2.1. General

9 Cue messages are mapped into DASH events, using inband `emsg` boxes and/or in-MPD events.  
10 Note that SCTE 35 cue message may not be sufficient by itself.

11 The examples below show use of SCTE 35 in user-defined events, and presentation time indicates  
12 the timing in within the Period.

13 Figure 14 below shows the content of an `emsg` box at the beginning of a segment with earliest  
14 presentation time  $T$ . There is a 6-sec warning of an upcoming splice – delta to splice time is indi-  
15 cated as 6 seconds – and duration is given as 1 minute. This means that an ad will start playing at  
16 time  $T + 6$  till  $T + 66$ . This example follows a practice defined in SCTE DVS 1208.

17

scheme_id_uri="urn:scte:scte35:2013:xml"
value=1001
timescale=90000
presentation_time_delta=540000
duration=5400000
id=0
message_data[]= <SpliceInfoSection ptsAdjustment="0" scte35:tier="22"> <SpliceInsert spliceEventId="111" spliceEventCancelIndicator="false" outOfNetworkIndicator="true" uniqueProgramId="65535" availNum="1" availsExpected="2" spliceImmediateFlag="false"> <Program><SpliceTime ptsTime="122342"/></Program> <BreakDuration autoReturn="false" duration="5400000"/> </SpliceInsert> <AvailDescriptor scte35:providerAvailId="332"/> </SpliceInfoSection>

18  
19

Figure 14 Inband carriage of SCTE 35 cue message

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19

Figure 15 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+cAAAAA==
      </scte35:Binary>
    </scte35:Signal>
  </Event>
</EventStream>
```

**Figure 15: 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.

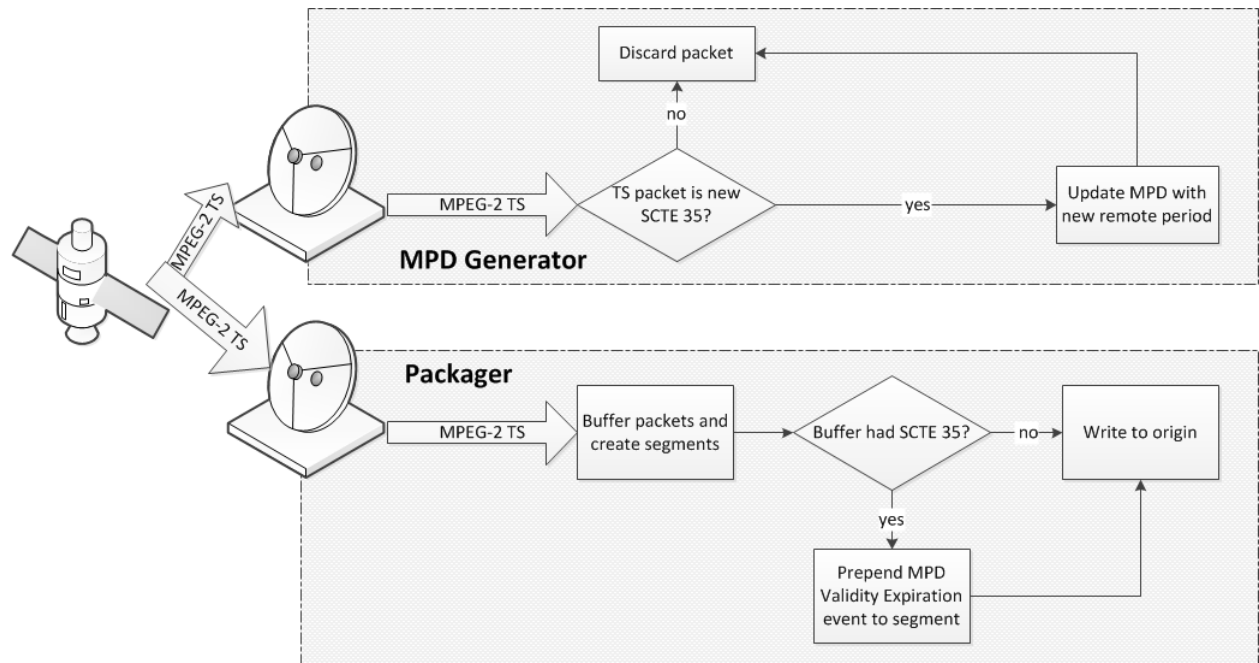
Normative definitions of carriage of SCTE 35 cue messages are in ANSI/SCTE 214-1 sec 6.8.4 (MPD) and ANSI/SCTE 214-3 sec 8.3.3.

**5.4.2.3. Asset Identifiers**

See sec. 5.3.2.2 for details.

**5.4.3. Workflows**

**5.4.3.1. Linear**



**Figure 16: 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).

---

1 Packager creates DASH segments out of the arriving feed and writes them into the origin server.  
2 The packager translates SCTE 35 cue messages into inband DASH events, which are inserted into  
3 media segments.  
4 MPD generator is unaware of ad insertion functionality and the packager does the translation of  
5 SCTE 35 cue messages into inband user-defined DASH events. On reception of an SCTE 35 cue  
6 message signaling an upcoming splice, a ``emsg`` with a translation of the cue message in its  
7 ``emsg`.message_data[]` field is inserted into the most recent Segment. This event triggers  
8 client interaction with an ad decision server, hence the sum of the earliest presentation time of the  
9 ``emsg`-bearing segment and the `emsg`.presentation_time_delta should be a trans-  
10 lation of splice_time() into the media timeline.  
11 An alternative implementation which is more compatible with server-based architecture in section  
12 5.3, an MPD generator can generate separate MPDs for both server-based and app-based architec-  
13 tures creating remote periods for server-based and in-MPD SCTE 35 events for app-based archi-  
14 tures, while a packager can insert inband MPD validity expiration events.  
15 A DASH client will pass the event to the app controlling it (e.g., via a callback registered by the  
16 app). The app will interpret the event and communicate with the ad decision server using some  
17 interface (e.g., VAST). This interface is out of the scope of this document.  
18 The communication with ad decision service will result in an MPD URL. An app will pause the  
19 presentation of the main content and start presentation of the inserted content. After presenting the  
20 inserted content the client will resume presentation of the main content. This assumes either proper  
21 conditioning of the main and inserted content or existence of separate client and decoder for in-  
22 serted content. The way pause/resume is implemented is internal to the API of the DASH client.  
23 Interoperability may be achieved by using the DASH MPD fragment interface, see ISO/IEC  
24 23009-1 [4], Annex C.4.`

#### 25 **5.4.3.2. On Demand**

26 As in the server-based case, functionality defined for the live case is sufficient. Moreover, the fact  
27 that that app-based implementation relies heavily on app's ability to pause and resume the DASH  
28 client, support for elastic workflows is provided out of the box.  
29 In the on demand case, as cue locations are well-known, it is advantageous to provide a static MPD  
30 with SCTE 35 events than run a dynamic service that relies on inband events.

## 31 **5.5. Extensions for ad insertion**

### 32 **5.5.1. Asset Identifiers**

33 **AssetIdentifier** descriptor shall be used for distinguishing parts of the same asset within a  
34 multi-period MPD, hence it shall be used for main content.  
35 In order to enable better tracking and reporting, unique IDs should be used for different assets.  
36 In the absence of other asset identifier schemes, a DASH-IF defined scheme may be used with the  
37 value of `@schemeIdUri` set to "urn:org:dashif:asset-id:2014". If used, the value  
38 of `@value` attribute descriptor shall be a MovieLabs ContentID URN ([59], 2.2.1) for the content.  
39 It shall be the same for all parts of an asset. Preferred schemes are EIDR (main content) and Ad-  
40 ID (advertising).  
41 If a Period has one-off semantics (i.e., an asset is completely contained in a single period, and its  
42 continuation is not expected in the future), the author shall not use asset identifier on these assets.

---

1 Periods that do not contain non-remote **AdaptationSet** elements, as well as zero-length peri-  
2 ods shall not contain the **AssetIdentifier** descriptor.

### 3 **5.5.2. Remote Periods**

4 An MPD may contain remote periods, some of which may have default content. Some of which  
5 are resolved into multiple Period elements.

6 After dereferencing MPD may contain zero-length periods or/and remote Periods.

7 In case of **Period@xlink:actuate="onRequest"**, MPD update and XLink resolution  
8 should be done sufficiently early to ensure that there are no artefacts due to insufficient time given  
9 to download the inserted content.

10 **Period@xlink:actuate="onRequest"** shall not be used if **MPD@type = "dynamic"**

### 11 **5.5.3. User-defined events**

#### 12 **5.5.3.1. Cue message**

13 Cue messages used in app-driven architecture shall be SCTE 35 events [55]. SCTE 35 event car-  
14 riage is defined in ANSI/SCTE 214-1 (MPD) and ANSI/SCTE 214-3 (inband). For MPD events,  
15 the XML schema is defined in SCTE 35 2014 [55] and allows either XML representation or con-  
16 cise base64-coded representation.

17 NOTE: PTS offset appearing in SCTE 35 shall be ignored, and only DASH event timing mechanism may be  
18 used to determine splice points.

#### 19 **5.5.3.2. Reporting**

20 MPD events with embedded IAB VAST 3.0 [54] response may be used for reporting purposes.

21 If only time-based reporting is required (e.g., reporting at start, completion, and quartiles), use of  
22 DASH callback event may be a simpler native way of implementing tracking. Callback events are  
23 defined in ISO/IEC 23009-1:2014 AMD3 [5].

#### 24 **5.5.3.3. Ad Insertion Event Streams**

25 Recommended Event Stream schemes along with their scheme identifier for app-driven ad inser-  
26 tion are:

27 1. "urn:scte:scte35:2013:bin" for inband SCTE 35 events containing a complete  
28 SCTE 35 section in binary form, as defined in ANSI/SCTE 214-3.

29 2. "urn:scte:scte35:2014:xml+bin" for MPD SCTE 35 events containing only  
30 base64 cue message representation, as defined in ANSI/SCTE 214-1.

31 NOTE: the content of Event element is an XML representation of the complete SCTE 35 cue mes-  
32 sages, that contains **Signal.Binary** element rather than the **Signal.SpliceInfoSection** ele-  
33 ment, both defined in SCTE 35 2014.

34 3. "http://dashif.org/identifiers/vast30" for MPD events containing  
35 VAST3.0 [54].

## 36 **5.6. Interoperability Aspects**

### 37 **5.6.1. Server-based Ad insertion**

38 For server-based ad insertion, the following aspects needs to be taken into account:

- 39 • Service offerings claiming conformance to server-based ad insertion shall follow the re-  
40 quirements 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 [9]. 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 [9] 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 [9] 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 [9].

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 [20]. 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 [20].
  - 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 [20].
  - 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 [20].
- 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 [10].
- Clients shall support H.264/AVC sample entries when SPS/PPS is provided in the Initialization Segment only according to ISO/IEC 14496-15, [10], i.e. sample entry 'avc1'.
- Clients shall support Inband Storage for SPS/PPS based ISO/IEC 14496-15, [10], 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 [9].
- The signaling of the different video codec profile and levels for the codecs parameters according to RFC6381 [11] is documented in Table 15. Note that any of the codecs present in Table 15 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 15 H.264 (AVC) Codecs parameter according to RFC6381 [11]**

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 [10]. 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 ISO BMFF and in DASH, in particular the use of the sync sample table and of the SAP sample group, please refer to Table 16.



1

**Table 16 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.**

2

- The signaling of the different video codec profile and levels for the codecs parameters is according to ISO/IEC 14496-15 [10] 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.

8

**Table 17 Codecs parameter according to ISO/IEC 14496-15 [10]**

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

---

1 that are exactly divisible subsamples of the source image size. As a result, the cropped vertical  
2 and horizontal sample counts of all Representations can be scaled to a common display size with-  
3 out position shift or aspect ratio distortion that would be visible during adaptive switching. Sub-  
4 sample ratios must result in integer values for the resulting encoded sample counts (without round-  
5 ing or truncation). The encoded sample count shall scale to the source video's exact active image  
6 aspect ratio when combined with the encoded sample aspect ratio value `aspect_ratio_idc`  
7 stored in the video Sequence Parameter Set NAL. Only the active video area shall be encoded so  
8 that devices can frame the height and width of the encoded video to the size and shape of their  
9 currently selected display area without extraneous padding in the decoded video, such as "letterbox  
10 bars" or "pillarbox bars".

11 All decoding parameter sets referenced by NALs in a Representation using 'avc1' or 'hvc1' sample  
12 description shall be indexed to that track's sample description table and decoder configuration  
13 record in the 'avcC' or 'hvcC' box contained in its Initialization Segment. All decoding param-  
14 eter sets referenced by NALs in a Representation using 'avc3' or 'hev1' sample description shall  
15 be indexed to a Sequence Parameter NAL (SPS) and Picture Parameter NAL (PPS) stored prior to  
16 the first video sample in that Media Segment. For 'avc3' and 'hev1' sample description Repre-  
17 sentations, the SPS and PPS NALs stored in 'avcC' or 'hvcC' in the Initialization Segment shall  
18 only be used for decoder and display initialization, and shall equal the highest Tier, Profile, and  
19 Level of any SPS in the Representation. SPS and PPS stored in each Segment shall be used for  
20 decoding and display scaling.

21 For all Representations within an Adaptation Set with the following parameters shall apply.

- 22 • All the Initialization Segments for Representations within an Adaptation Set shall have the  
23 same sample description `codingname`. For example the inclusion of 'avc1' and 'avc3'  
24 based Representations within an Adaptation Set or the inclusion 'avc1' and 'hev1' based  
25 Representations within an Adaptation Set is not permitted.
- 26 • All Representations shall have equal timescale values in all `@timescale` attributes and  
27 'tkhd' `timescale` fields in Initialization Segments.
- 28 • If 'avc1' or 'hvc1' sample description is signaled in the **AdaptationSet**`@codecs`  
29 attribute, an edit list may be used to synchronize all Representations to the presentation  
30 timeline, and the edit offset value shall be equal for all Representations.
- 31 • Representations in one Adaptation Set shall not differ in any of the following parameters:  
32 Color Primaries, Transfer Characteristics and Matrix Coefficients. If Adaptation Sets differ  
33 in any of the above parameters, these parameters should be signaled on Adaptation Set  
34 level. If signaled, a Supplemental or Essential Property descriptor shall be used, with the  
35 `@schemeIdURI` set to `urn:mpeg:mpegB:cicp:<Parameter>` as defined in  
36 ISO/IEC 23001-8 [50] and `<Parameter>` one of the following: `ColourPrimaries`,  
37 `TransferCharacteristics`, or `MatrixCoefficients`. The `@value` attribute  
38 shall be set as defined in ISO/IEC 23001-8 [50].

#### 39 6.2.5.2. Bitstream Switching

40 For AVC and HEVC video data, if the `@bitstreamswitching` flag is set to true, then the  
41 following additional constraints shall apply:

- 42 • All Representations shall be encoded using 'avc3' sample description for AVC or  
43 'hev1' for HEVC, and all IDR pictures shall be preceded by any SPS and PPS NAL  
44 decoding parameter referenced by a video NAL in that codec video sequence.

---

1  
2 Note: NAL parameter indexes in a Media Segment are scoped to that Segment. NALs and  
3 indexes in the Initialization Segment may be different, and are only used for decoder ini-  
4 tialization, not Segment decoding.

- 5 • All Representations within a video Adaptation Set shall include an Initialization Segment  
6 containing an ‘avcC’ or ‘hvcC’ Box containing a Decoder Configuration Record con-  
7 taining SPS and PPS NALs that equal the highest Tier, Profile, Level, vertical and hori-  
8 zontal sample count of any Media Segment in the Representation. HEVC Decoder Con-  
9 figuration Records shall also include a VPS NAL.
- 10 • The **AdaptationSet@codecs** attribute shall be present and equal the maximum pro-  
11 file and level of any Representation contained in the Adaptation Set.
- 12 • The **Representation@codecs** attribute may be present and in that case shall equal  
13 the maximum profile and level of any Segment in the Representation.
- 14 • Edit lists shall not be used to synchronize video to audio and presentation timelines.
- 15 • Video Media Segments shall set the first presented sample’s composition time equal to  
16 the first decoded sample’s decode time, which equals the `baseMediaDecodeTime` in  
17 the Track Fragment Decode Time Box (‘`tfdt`’).

18  
19 Note: This requires the use of negative composition offsets in a v1 Track Run Box  
20 (‘`trun`’) for video samples, otherwise video sample reordering will result in a delay of  
21 video relative to audio.

- 22 • The **AdaptationSet@presentationTimeOffset** attribute shall be sufficient to  
23 align audio video, subtitle, and presentation timelines at presentation a Period’s presenta-  
24 tion start time. Any edit lists present in Initialization Segments shall be ignored. It is  
25 strongly recommended that the Presentation Time Offset at the start of each Period coin-  
26 cide with the first frame of a Segment to improve decoding continuity at the start of Peri-  
27 ods.

28 NOTE: An Adaptation Set with the attribute **AdaptationSet@bitstreamSwitching="true"** fulfills  
29 the requirements of the DVB DASH specification [43].

30 See section 7.7 for additional Adaptation Set constraints related to content protection.

## 31 **6.3. Audio**

### 32 **6.3.1. General**

33 Content offered according to DASH-IF IOP is expected to contain an audio component in most  
34 cases. Therefore, clients consuming DASH-IF IOP-based content are expected to support stereo  
35 audio. Multichannel audio support and support for additional codecs is defined in extensions in  
36 section 9 of this document.

37 The codec for basic stereo audio support is MPEG-4 High Efficiency AAC v2 Profile, level 2 [12].

38 Notes

- 39 • HE-AACv2 is also standardized as Enhanced aacPlus in 3GPP TS 26.401 [14].
- 40 • HE-AACv2 Profile decoder [9] can also decode any content that conforms to
  - 41 ○ MPEG-4 AAC Profile [12]
  - 42 ○ MPEG-4 HE-AAC Profile [12]

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 [43], 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 [7] 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 [11] and is documented in Table 18. Table 18 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 18 HE-AACv2 Codecs parameter according to RFC6381 [11]

Codec	Codec Parameter	ISO BMFF Encapsulation	SAP type
MPEG-4 AAC Profile [12]	mp4a.40.2	ISO/IEC 14496-14 [13]	1
MPEG-4 HE-AAC Profile [12]	mp4a.40.5	ISO/IEC 14496-14 [13]	1
MPEG-4 HE-AAC v2 Profile [12]	mp4a.40.29	ISO/IEC 14496-14 [13]	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:

- 
- 1 • the `@audioSamplingRate` attribute for signaling the sampling rate of the audio media  
2 component type in section 5.3.7 of ISO/IEC 23009-1
  - 3 • the **AudioChannelConfiguration** element for signaling audio channel configura-  
4 tion of the audio media component type in section 5.3.7 of ISO/IEC 23009-1.

## 5 **6.4. Auxiliary Components**

### 6 **6.4.1. Introduction**

7 Beyond regular audio and video support, TV programs typically also require support for auxiliary  
8 components such as subtitles and closed captioning, often due to regulatory requirements. DASH-  
9 IF IOP provides tools to address these requirements.

### 10 **6.4.2. Subtitles and Closed Captioning**

11 Technologies for subtitles are as follows:

- 12 • CEA-708 Digital Television (DTV) Closed Captioning [15]
- 13 • IMSC1 [62] conformant profiles of TTML, packaged as Segments conforming to MPEG-  
14 4, Part 30 [30], including subsets such as:
  - 15 ○ W3C TTML [17]
  - 16 ○ SMPTE Timed Text [18] (including image-based subtitles and closed captioning)
  - 17 ○ EBU-TT [21]
- 18 • 3GPP Timed Text [16]
- 19 • Web VTT [19]

20 For simple use cases, CEA-608/708 based signaling as defined in section 6.4.3 may be used.

21 For any other use cases, IMSC1 [62] should be used as defined in section 6.4.4. It is expected that  
22 most subset profiles of IMSC1 would be reasonably decodable.

23 TTML and WebVTT Media Segments shall be referenced by Representation elements in MPDs,  
24 downloaded, initialized, and synchronized for multimedia presentation the same as audio and  
25 video Segments.

26 Note: DASH playback applications such as Web pages can download TTML or WebVTT text files, initialize  
27 renderers, and synchronize rendering and composition. This specification does not specify interoperable play-  
28 back of these “sidecar” subtitle files in combination with a DASH audio visual presentation.

### 29 **6.4.3. CEA-608/708 in SEI messages**

#### 30 **6.4.3.1. Background**

31 In order to provide the signaling of the presence of SEI-based data streams and closed captioning  
32 services on MPD level, descriptors on DASH level are defined. This section provides some back-  
33 ground.

34 Note: This method is compatible with draft SCTE specification DVS 1208 and therefore  
35 SCTE URNs are used for the descriptor `@schemeIdURI`. In an updated version of this  
36 document more details on the exact relation to the SCTE specification will be provided.

37 The presence of captions and their carriage within the SEI message of a video track is defined in  
38 ANSI/SCTE 128-1 2013 [44], section 8.1 Encoding and transport of caption, active format de-  
39 scription (AFD) and bar data.

40 Based on this it is enabled that a video track carries SEI message that carry CEA-608/708 CC. The  
41 SEI message `payload_type=4` is used to indicate that Rec. ITU-T T.35 based SEI messages  
42 are in use.

---

1 In summary the following is included in ANSI/SCTE 128-1 2013 to signal CEA-608/708 CC:

- 2 • `SEI payloadType` is set to 4
- 3 • `itu_t_t35_country_code` – A fixed 8-bit field, the value of which shall be 0xB5.
- 4 • `itu_t_35_provider_code` – A fixed 16-bit field registered by the ATSC. The value
- 5 shall be 0x0031.
- 6 • `user_identifier` – This is a 32 bit code that indicates the contents of the
- 7 `user_structure()` and is 0x47413934 (“GA94”).
- 8 • `user_structure()` – This is a variable length data structure `ATSC1_data()` defined
- 9 in section 8.2 of ANSI/SCTE 128 2013-a.
- 10 • `user_data_type_code` is set to 0x03 for indicating captioning data in the
- 11 `user_data_type_structure()`
- 12 • `user_data_type_structure()` is defined in section 8.2.2 of ANSI/SCTE 128-1
- 13 2013 for Closed Captioning and defines the details on how to encapsulate the captioning
- 14 data.

15 The semantics of relevant Caption Service Metadata is provided in CEA-708 [15], section 4.5:

- 16 • the total number of caption services (1-16) present over some transport-specific period.
- 17 • For each service:
  - 18 ○ The type of the service, i.e. being 608 or 708. According to CEA-708 [15], section
  - 19 4.5, there shall be at most one CEA-608 data stream signaled. The CEA-608
  - 20 datstream itself signals the individual CEA-608-E caption channels.
  - 21 ○ When the type of the service is 708, then the following 708-related metadata
  - 22 should be conveyed:
    - 23 ▪ **SERVICE NUMBER**: the service number as found on the 708 caption ser-
    - 24 vice block header (1-31). This field provides the linkage of the remaining
    - 25 metadata to a specific 708 caption service
    - 26 ▪ **LANGUAGE**: the dominant language of the caption service, recommended
    - 27 to be encoded from ISO 639.2/B [46].
    - 28 ▪ **DISPLAY ASPECT RATIO {4:3, 16:9}**: The display aspect ratio assumed
    - 29 by the caption authoring in formatting the caption windows and contents.
    - 30 ▪ **EASY READER**: this metadata item, when present, indicates that the ser-
    - 31 vice contains text tailored to the needs of beginning readers.

#### 32 6.4.3.2. MPD-based Signaling of SEI-based CEA-608/708 Closed Caption services

33 This subsection provides methods MPD-based Signaling of SEI-based CEA-608/708 Closed Cap-

34 tion services, i.e.

- 35 • The presence of one or several SEI-based closed caption services in a Representation.
- 36 • The signaling of the relevant Caption Service Metadata as defined in CEA-708 [15], sec-
- 37 tion 4.5.

38 The descriptor mechanism in DASH is used for this purpose.

39 Signaling is provided by including **Accessibility** descriptors, one each for CEA 608 and

40 CEA 708 and is described in sections 6.4.3.3 and 6.4.3.4, respectively. The **Accessibility**

41 descriptor is included for the **AdaptationSet** and all included Representations shall provide

42 equivalent captions.

1 The @value attribute of each descriptor can be either list of languages or a complete map of  
2 services (or CC channels, in CEA-608 terminology). Listing languages without service or channel  
3 information is strongly discouraged if more than one caption service is present.

4 These definitions are equivalent to SCTE 214-1 [57].

### 5 **6.4.3.3. Signaling CEA-608 caption service metadata**

6 The **Accessibility** descriptor shall be provided with @schemeIdURI set to  
7 urn:scte:dash:cc:cea-608:2015, and an optional @value attribute to describe the cap-  
8 tions. If the @value attribute is not present, the Representation contains a CEA-608 based closed  
9 captioning service.

10 If present, the @value attribute shall contain a description of caption service(s) provided in the  
11 stream as a list of channel-language pairs. Alternatively, a simple list of language codes may be  
12 provided, but this is strongly discouraged as it will not provide sufficient information to map the  
13 language with the appropriate caption channel.

14 The @value syntax shall be as described in the ABNF below.

```
15 @value          = (channel *3 [";" channel]) / (language *3[";" language])  
16 channel         = channel-number "=" language  
17 channel-number  = CC1 | CC2 | CC3 | CC4  
18 language        = 3ALPHA ; language code per ISO 639.2/B [46]
```

### 19 **6.4.3.4. Signaling CEA-708 caption service metadata**

20 The **Accessibility** descriptor shall be provided with the @schemeIdURI set to  
21 urn:scte:dash:cc:cea-708:2015, and an optional @value attribute to describe the  
22 captions. If the @value attribute is not present, the Representation contains a CEA-708 based  
23 closed captioning service.

24 If present, the @value shall contain the Caption Service Metadata as provided in CEA-708 [15],  
25 section 4.5 as a semicolon-separated string of service descriptions. Each service description is a  
26 list of colon-separated name-value pairs. Alternatively, a simple list of language codes may be  
27 provided, but this is strongly discouraged as it will not provide sufficient information to map the  
28 language with the appropriate caption service. The @value syntax shall be as described in the  
29 ABNF below.

```
30 @value          = service *15 [";" service] / (language *15[";" language])  
31 service         = service-number "=" param  
32 service-number  = (%d1 - %d63) ; decimal numbers 1 through 63  
33 param          = "lang" ":" language[";", "easy-reader"] [";", "aspect-ratio"]  
34 language        = 3ALPHA; language code per ISO 639.2/B  
35 easy-reader     = "er" ":" BIT ; default value 0  
36 wide-aspect-ratio = "war" ":" BIT ; default value 1 (16:9), 0 for 4:3
```

37 NOTE: ALPHA and BIT are as defined by IETF RFC 5234 [48], Appendix B.1.

38

39 Based on the above ABNF, the following parameters are defined for signaling CEA-708 metadata:

Name	Value	Comments
lang	LANG	LANGUAGE code (see above)

war	Aspect ratio of the service.	The value may be either "4:3" (set to 0) or "16:9" (set to 1).
er	EasyReader presence ('1' if present, '0' if not).	If not present, 0 is assumed

1 Note that each of the `service` parameters (except for language) may be present or not present.  
2 Default values can be assumed where specified.

### 3 6.4.3.5. Examples

4 Simple signaling of presence of CEA-608 based closed caption service (Note: Not signaling lan-  
5 guages is a discouraged practice)

```
6 <Accessibility
7   schemeIdUri="urn:scte:dash:cc:cea-608:2015"/>
```

8 Signaling of presence of CEA-608 closed caption service languages in English and German

```
9 <Accessibility
10   schemeIdUri="urn:scte:dash:cc:cea-608:2015"
11   value="eng;deu"/>
```

12 Signaling of presence of CEA-608 closed caption service in English and German, with channel  
13 assignments

```
14 <Accessibility
15   schemeIdUri="urn:scte:dash:cc:cea-608:2015"
16   value="CC1=eng;CC3=deu"/>
```

17 Signaling of presence of CEA-708 closed caption service in English and German

```
18 <Accessibility
19   schemeIdUri="urn:scte:dash:cc:cea-708:2015"
20   value="1=lang:eng;2=lang:deu"/>
```

21 Signaling of presence of CEA-708 closed caption service in English and easy reader English

```
22 <Accessibility
23   schemeIdUri="urn:scte:dash:cc:cea-708:2015"
24   value="1=lang:eng;2=lang:eng,war:1,er:1"/>
```

### 25 6.4.4. Timed Text (IMSC1)

26 W3C TTML [17] and its various profiles - W3C IMSC1 (text and image profiles) [62], SMPTE  
27 Timed Text [18], and EBU Timed Text [21] - provide a rich feature set for subtitles. Beyond basic  
28 subtitles and closed captioning, for example, graphics-based subtitles and closed captioning are  
29 also supported by IMSC1. Conversion of CEA-608 and CEA-708 into IMSC1 may be done ac-  
30 cording to SMPTE 2052-10 [28] and SMPTE-2052-11 [29], respectively. The Timed Text track  
31 shall conform to IMSC1 [62]. Note that by the choice of IMSC1 as the supported format at the  
32 client, other formats such as EBU TT [21] are also supported because they are subset profiles.

33 In the context of DASH, the following applies for text/subtitling:

- 34 • All graphics type samples shall be SAP type 1.
- 35 • The signalling of the different text/subtitling codecs for the codecs parameters is according  
36 to W3C TTML Profile Registry [63] and is documented in Table 19. Table 19 also pro-  
37 vides information on ISO BMFF encapsulation.
- 38 • For live services, encapsulation shall be in ISO BMFF. However, for On-Demand cases,  
39 the full file of subtitles may be provided in either ISO BMFF or XML data only.

40 **Table 19 Subtitle MIME type and codecs parameter according to IANA and W3C registries**

Codec	MIME type	Codecs Pa- rameter	ISO BMFF Encapsula- tion
-------	-----------	-----------------------	-----------------------------



		@codecs	
<b>IMSC1 Timed Text [62] without encapsulation</b>	application/ttml+xml <sup>(1,3)</sup>	See [63]	n/a
<b>IMSC1 Timed Text [62] with ISO BMFF encapsulation</b>	application/mp4	See [63]	ISO/IEC 14496-12 [8] ISO/IEC 14496-30 [30]
Notes:			
(1) DVB DASH only supports ISO BMFF encapsulated TT, but not XML-based.			

1

## 2 **6.4.5. Annotation of Subtitles**

3 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  
4 role scheme may be used. Guidelines for annotation are for example provided in DVB DASH,  
5 section 7.1.2 or SCTE 214-1 [57], section 7.2.  
6

# 7 **7. Content Protection and Security**

## 8 **7.1. Introduction**

9 DASH-IF IOPs do not intend to specify a full end-to-end DRM system. However DASH-IF IOP  
10 provides a framework for multiple DRMs to protect DASH content by adding instructions or *Pro-*  
11 *tection System Specific*, proprietary information in predetermined locations in MPDs, or DASH  
12 content that is encrypted with Common Encryption as defined in ISO/IEC 23001-7 [31].

13 The Common Encryption ('cenc') protection scheme specifies encryption parameters that can be  
14 applied by a scrambling system and key mapping methods using a common key identifier (KID)  
15 to be used by different DRM systems such that the same encrypted version of a file can be com-  
16 bined with different DRM systems that can store proprietary information for licensing and key  
17 retrieval in the Protection System Specific Header Box ('pssh'), or in **ContentProtection**  
18 Descriptors in an MPD. The DRM scheme for each pssh is identified by a DRM specific Sys-  
19 temID.

20 The recommendations in this document reduce the encryption parameters and use of the encryption  
21 metadata to specific use cases for VOD and live content with key rotation.

22 The base technologies are introduced first followed by informative chapter on standardized ele-  
23 ments. Additional Content Protection Constraints are then listed that are specific to conformance  
24 to DASH-264/AVC IOP.

## 25 **7.2. HTTPS and DASH**

26 The use of HTTP within DASH as a transport protocol, especially for retrieving media segments,  
27 inherently provides many advanced features such as caching and redirection. However, when not  
28 encrypted, data delivered over HTTP are in the clear, and there may not be any guarantee that the

---

1 clear-text data are not being eavesdropped and tampered. If this is of concern, transport security in  
2 HTTP-based delivery may be achieved by using HTTP over TLS (HTTPS) as specified in RFC  
3 2818. HTTPS is a protocol for secure communication over a computer network which is widely  
4 used on the Internet and also increasingly used for content streaming, mainly for the following  
5 purposes:

- 6 • authenticating the streaming server with which a streaming client is communicating,
- 7 • protecting the privacy of the exchanged data from eavesdropping by providing encryption  
8 of bidirectional communications between a client and a server, and
- 9 • ensuring integrity of the exchanged data against man-in-the-middle attacks against tampering  
10 with and/or forging the contents of the communication.

11 Many HTTP streaming technologies (as well as many web sites) are moving to HTTPS delivery,  
12 such as Netflix, Youtube, Facebook (see for example presentations at  
13 <http://dashif.org/events/event-2015-08-20/>). One of the main reasons is that with Cross Origin Re-  
14 source Sharing (CORS) (<http://www.w3.org/TR/cors/>). As all web browsers and services (such as  
15 Facebook and Google) move to HTTPS and CORS does not allow that if the origin web page is  
16 HTTPS that you move to HTTP for resources delivered as links in the web page, the media also  
17 needs to be delivered through HTTPS.

18 MPEG-DASH explicitly permits the use of https as a scheme and hence, HTTP over TLS as a  
19 transport protocol. When using HTTPS in DASH, one can for instance specify that all media seg-  
20 ments are delivered over HTTPS, by declaring that all the `<BaseURL>`'s are HTTPS based, as follow:

```
21 <BaseURL>https://cdn1.example.com/</BaseURL>  
22 <BaseURL>https://cdn2.example.com/</BaseURL>
```

23 One can also use HTTPS for retrieving other types of data carried with an MPD that are HTTP-  
24 URL based, such as DRM licenses specified within the `<ContentProtection>` descriptor:

```
25 <ContentProtection schemeIdUri="http://example.net/052011/drm">  
26 <drm:License>https://MoviesSP.example.com/protect?license=kljkl sdfiowek</drm:License>  
27 </ContentProtection>
```

28 Because of the CORS recommendation of not moving from HTTPS requests to HTTP ones, when-  
29 ever HTTPS is used for any portion of DASH content referenced within an MPD, it is recom-  
30 mended that HTTPS be used for the entirety of the DASH content referenced in the same MPD.

31 While using HTTPS in DASH provides good levels of trust and authenticity for data exchanged  
32 between DASH servers and clients connected over HTTPS, it should be pointed out that HTTPS  
33 only protects the transport link, but not the access to streaming content and the usage of streamed  
34 content. HTTPS itself does not imply user authentication and content authorization (or access con-  
35 trol). This is especially the case that HTTPS provides no protection to any streamed content cached  
36 in a local buffer at a client for playback. For this reason, using HTTPS does not replace the need  
37 for content protection and digital rights management of streaming content.

38 There are also some impacts of using HTTPS in DASH that are going to be challenges in main-  
39 taining DASH advantages and end-user streaming experiences:

- 40 • CDN: Because of HTTPS encrypting the connection between a client and a server for each  
41 HTTPS session, it causes difficulties in caching encrypted DASH segment content on  
42 CDNs to reuse it for other sessions and other clients.

- 
- 1 • Network: As use of HTTPS establishes a secure tunnel between a client and a server, it is  
2 impossible for underlying networks (or any entities on the data delivery path between the  
3 client and server) to manage and optimize encrypted data traffics over the networks, espe-  
4 cially when the networks are mobile, based on inspecting and analyzing content of the  
5 traffics.
  - 6 • Efficiency: Using HTTPS introduces "double encryption" when streaming content is al-  
7 ready encrypted, for instance, using a content protection or DRM system. While delivering  
8 encrypted content over HTTPS may not increase content security, it does add extra encryp-  
9 tion overhead and therefore latency in streaming encrypted content.

### 10 7.3. Base Technologies Summary

11 The normative standard that defines common encryption in combination with ISO BMFF is  
12 ISO/IEC 23001-7 [31]. It includes:

- 13 • Common ENCRyption (CENC) of NAL structure video and other media data with AES-  
14 128 CTR mode
- 15 • Support for decryption of a single Representation by multiple DRM systems
- 16 • Key rotation (changing media keys over time)
- 17 • XML syntax for expressing a default KID attribute and pssh element in MPDs

18 The main DRM components are:

- 19 1. The **ContentProtection** descriptors in the MPD (see [5] 5.3.7.2-Table 9, 5.8.5.2 and  
20 [5] 5.8.4.1) that contains the URI for signaling of the use of Common Encryption or the  
21 specific DRM being used.
- 22 2. 'tenc' parameters that specify encryption parameters and default\_KID (see [31] 8.2).  
23 The 'tenc' information is in the Initialization Segment. Any KIDs in Movie Fragment  
24 sample group description boxes override the 'tenc' parameter of the default\_KID, as  
25 well as the 'not encrypted' parameter. Keys referenced by KID in sample group descrip-  
26 tions must be available when samples are available for decryption, and may be stored in a  
27 protection system specific header box ('pssh') in each movie fragment box ('moof').  
28 The default\_KID information may also appear in the MPD (see [31] 11).
- 29 3. 'senc' parameters that may store initialization vectors and subsample encryption ranges.  
30 The 'senc' box is stored in each track fragment box ('traf') of an encrypted track (see  
31 [31] 7.1), and the stored parameters accessed using the sample auxiliary information offset  
32 box ('sai0') and the sample auxiliary information size box ('sais') (see [4] 8.7.8 and  
33 8.7.9).
- 34 4. 'pssh' license acquisition data or keys for each DRM in a format that is "Protection Sys-  
35 tem Specific". 'pssh' refers to the Protection System Specific Header box described in  
36 [31], 8.1.2. 'pssh' boxes may be stored in Initialization or Media Segments (see [31]  
37 8.1 and 8.2). It may also be present in a cenc:pssh element in the MPD (see [5] 5.8.4.1,  
38 [31] 11.2.1). cenc:pssh information in the MPD allows faster parsing, earlier access,  
39 identification of duplicate license requests, and addition of DRMs without content modifi-  
40 cation. 'pssh' boxes in Initialization Segments are not recommended because they trigger  
41 a license request each time an Initialization Segment is processed in a Web browser for  
42 each Representation and bitrate switch.

---

1 *Note: The duplication of the pssh information in the Initialization Segment may cause*  
2 *difficulties in playback with HTML5 - EME based players. I.e. content will fail unless play-*  
3 *ers build complex DRM specific license handling.*

4 5. Key rotation is mainly used to allow changes in entitlement for continuous live content. It  
5 is used as defined in [31] with the following requirements:

- 6 • Sample To Group Box ( 'sbgp' ) and Sample Group Description Box ( 'sgpd' )  
7 of type 'seig' are used to indicate the KID applied to each sample, and changes  
8 to KIDs over time (i.e. "key rotation"). (see [4] 8.9.4) KIDs referenced by sample  
9 groups must have the keys corresponding to those KIDs available when the sam-  
10 ples in a Segment are available for decryption. Keys referenced by sample groups  
11 in a Segment may be stored in that Segment in Protection System Specific Header  
12 Boxes ( 'pssh' ) stored in the Movie Fragment Box ( 'moof' ). A version 1  
13 'pssh' box may be used to list the KID values stored to enable removal of  
14 duplicate boxes if a file is defragmented.
- 15 • Keys stored in Media Segment 'pssh' boxes must be stored in the same DRM  
16 format for all users so that the same Media Segments can be shared by all users.  
17 User-specific information must be delivered "out of band", as in a "root" license  
18 associated with the default\_KID, which can be individualized for each DRM  
19 client, and control access to the shared 'pssh' information stored in Media Seg-  
20 ments, e.g. by encrypting the keys stored in Segment 'pssh' boxes with a "root  
21 key" provided by the user-specific DRM root license. Common Encryption speci-  
22 fies 'pssh' to enable key storage in movie fragments/Segments; but it does not  
23 preclude other methods of key delivery that satisfy KID indexing and availability  
24 requirements.
- 25 • For details see Section 7.5.

## 26 **7.4. ISO BMFF Support for Common Encryption and DRM**

### 27 **7.4.1. Box Hierarchy**

28 The ISO Media Format carries content protection information in different locations. Their hierar-  
29 chy is explained in the informational chapter below, followed by a reference on where these ele-  
30 ments are standardized.

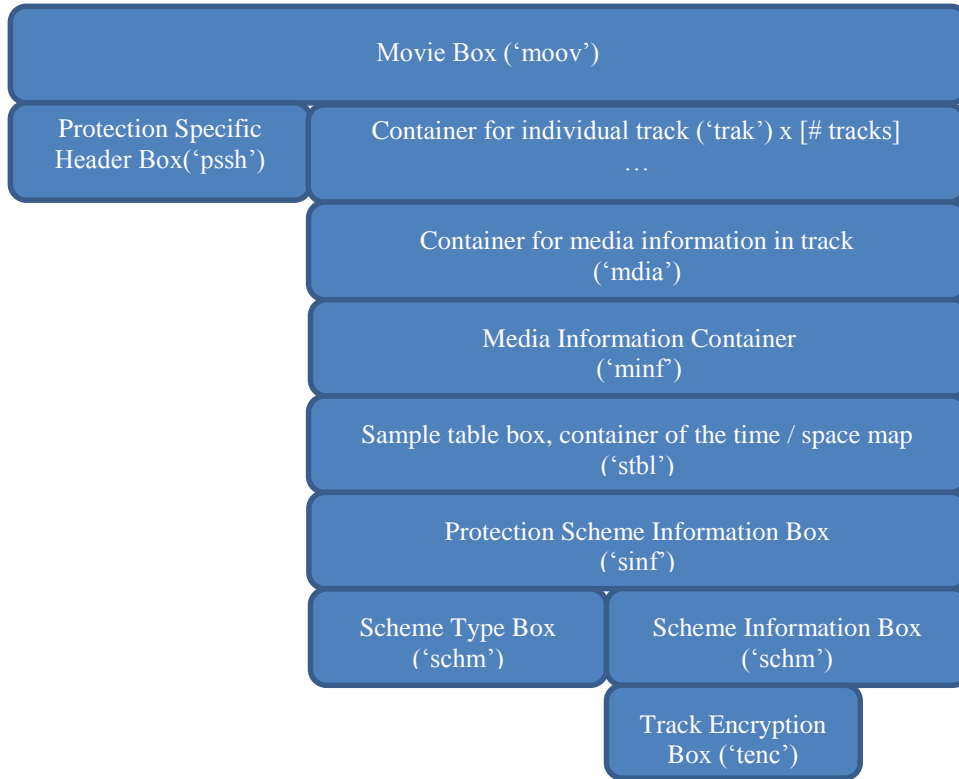
31 The following shows the box hierarchy and composition for relevant boxes, when using common  
32 encryption:

- 34 • moov/pssh (zero or one per system ID)
- 35 • moov/trak/mdia/minf/stbl/stsd/sinf/schm (one, if encrypted)
- 36 • moov/trak/mdia/minf/stbl/stsd/sinf/schi/tenc (one, if encrypted)
- 37 • moof/traf/saiz (one, if encrypted)
- 38 • moof/traf/saio (one, if encrypted)
- 39 • moof/traf/senc (one, if encrypted)

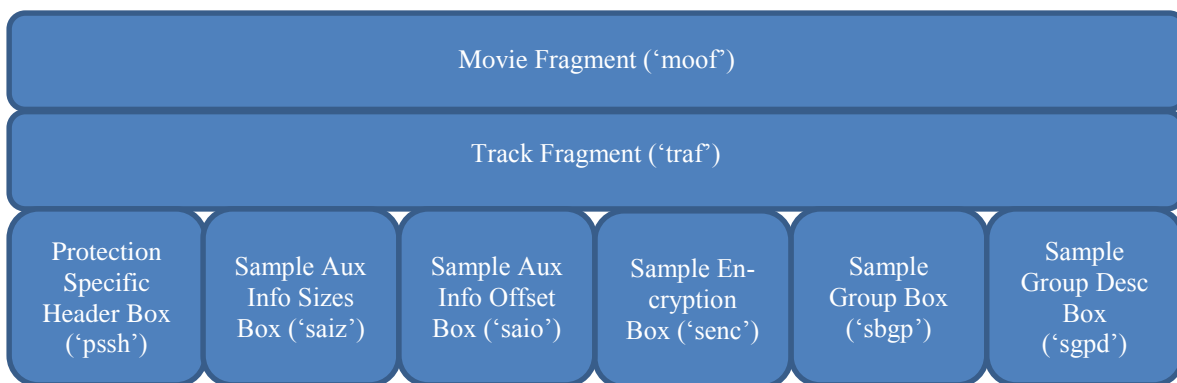
40 for key rotation

- 1 • moof/traf/sbgp (one per sample group)
- 2 • moof/traf/sgpd 'seig' (sample group entry) (one per sample group)
- 3 • moof/pssh (zero or one per system ID)

4 Graphical overviews of above structure for VOD content and live content are shown in Figure 17  
 5 and Figure 18 respectively.



6  
 7 **Figure 17: Visualization of box structure for single key content**



9  
 10 **Figure 18: Visualization of box structure with key rotation**

## 1 7.4.2. ISO BMFF Structure Overview

2 Table 20 provides pointers to relevant information in the specifications to understand the standard  
 3 DRM components and if the main description is located in the ISO base media file format ([8]),  
 4 or the Common Encryption specification ([31]).

5 **Table 20 Boxes relevant for DRM systems**

Box	Full Name / Usage	Reference
moof	movie fragment header <i>One 'moof' box for each fragment, i.e. Media Segment/Subsegment.</i>	[8] 8.32 + [5]
moov	movie header, container for metadata <i>One 'moov' box per file.</i>	[8] 8.1
pssh	Protection System Specific Header Box <i>Contains DRM specific data. pssh box version 1 (specified in Common Encryption 2<sup>nd</sup> edition) contains a list of KIDs to allow removing duplicate 'pssh' boxes when defragmenting a file by comparing their KIDs</i>	[31] 8.1.1
saio	Sample Auxiliary Information Offsets Box <i>Contains the offset to the IVs &amp; subsample encryption byte ranges.</i>	[8] 8.7.9
sais	Sample Auxiliary Information Sizes Box <i>Contains the size of the IVs &amp; subsample encryption byte ranges.</i>	[8] 8.7.8
senc	Sample Encryption Box <i>Contains Initialization Vectors; and subsample ranges for a Media Segment</i>	[31] 7.1
schi	Scheme Information Box <i>Container boxes used by that protection scheme type.</i>	[8] 8.12.6 + [31] 4
schm	Scheme Type Box <i>Contains the encryption scheme, identified by a 4 character code, e.g. 'cenc'</i>	[8], 8.12.5 + [31] 4
seig	Cenc Sample Encryption Information Video Group Entry <i>A sample description containing KIDs describing sample groups in this segment, for key rotation.</i>	[31] 6
sbgp	Sample to Group Box <i>lists a group of samples</i>	[8] +[31] 5
sgpd	Sample Group Description Box <i>Describes properties of a sample group</i>	[8] 8.9.3 + [31] 5
sinf	Protection Scheme Information Box <i>Signals that the stream is encrypted</i>	[8] 8.12.1 + [31] 4
stsd	Sample description table (codec type, initialization parameters, stream layout, etc.)	[8] 8.16
tenc	Track Encryption Box <i>Contains default encryption parameters for the entire track, e.g. default_KID</i>	[31] 8.2.1

6

---

## 1 **7.5. Periodic Re-Authorization**

### 2 **7.5.1. Introduction**

3 This section explains different options and tradeoffs to enable change in keys (aka key rotation),  
4 considering different use cases, application scenarios, content encoding variants and signaling re-  
5 quirements.

### 6 **7.5.2. Use Cases and Requirements**

7 The main use case in this context is to enable service changes at program boundaries, not to in-  
8 crease security of CENC by preventing e.g. key factoring or key redistribution. In order to clarify  
9 this application, the term *periodic re-authorization* is used instead of the term *key rotation*.

10 In addition, this is one of the ways to implement counting of active streams as they are periodically  
11 requesting keys from a license server.

12 The following use cases and requirements have been considered:

- 13 • Ability to force a client device to re-authorize to verify that it is still authorized for con-  
14 tent consumption.
- 15 • Support for distribution models such as: Live content, PVR, PPV, VOD, SVOD, live to  
16 VOD, network DVR. This includes where live content is converted into another con-  
17 sumption license for e.g. catch up TV.
- 18 • Uninterrupted playback when keys are rotated.
  - 19 ○ Preventing of client storm: Requests from client should be distributed where pos-  
20 sible to prevent spiking loads at isolated times.
  - 21 ○ Quick recovery: If the server or many client devices fail, the service should be  
22 able to resume quickly.
  - 23 ○ Player visibility into the key rotation signal
- 24 • Regional blackout: Device location may be taken into account to enable de-activation of  
25 content in a geographical area.
- 26 • Hybrid broadcast/unicast networks in which receivers operating in broadcast-only mode at  
27 least some of the time, i.e. unable to always download licenses on-demand through unicast.
- 28 • No required changes to the standard process and validity of MPDs.

### 29 **7.5.3. Implementation Options**

#### 30 **7.5.3.1. General**

31 This section describes approaches for *periodic re-authorization*; recommended because they best  
32 cover the use cases and allow interoperable implementation. Other approaches are possible and  
33 may be considered by individual implementers.

34 One of those is explicit signaling using e.g. `esm` messages, using a custom key rotation signal to  
35 indicate future KIDs.





---

1 bound to one DRM domain associated with one user ID and, and enables access to ECLs and media  
2 keys associated with each show it authorizes.

3 Entitlement Control License (ECL) – A license that contains a media key and can only be accessed  
4 by provisioned devices that have been authorized by installing the associated EML. ECLs may be  
5 delivered with the media in a broadcast distribution.

6 Changing media keys and ECLs per asset, forces re-authorization of each show by the DRM sys-  
7 tem that must extract the media key.

## 8 **7.6. MPD support for Encryption and DRM Signaling**

### 9 **7.6.1. Introduction**

10 The MPD contains signaling of the content encryption and key management methods used to  
11 help the receiving client determine if it can possibly play back the content. The MPD elements to  
12 be used are the **ContentProtection** Descriptor elements. At least one **Content Protec-**  
13 **tion** Descriptor element SHALL be present in each **AdaptationSet** element describing en-  
14 crypted content.

### 15 **7.6.2. Use of the Content Protection Descriptor**

#### 16 **7.6.2.1. ContentProtection Descriptor for *mp4protection* Scheme**

17 A **ContentProtection** descriptor with the @schemeIdUri value equals to  
18 "urn:mpeg:dash:mp4protection:2011" signals that content is encrypted with the  
19 scheme indicated in the @value attribute. The file structure of content protection schemes is  
20 specified in [8], 5.8.5.2, and the @value = 'cenc' for the Common Encryption scheme, as spec-  
21 ified in [31]. Although the **ContentProtection** Descriptor for UUID Scheme described be-  
22 low is usually used for license acquisition, the **ContentProtection** Descriptor with  
23 @schemeIdUri="urn:mpeg:dash:mp4protection:2011" and with @cenc:de-  
24 fault\_KID may be sufficient to acquire a license or identify a previously acquired license that  
25 can be used to decrypt the Adaptation Set. It may also be sufficient to identify encrypted content  
26 in the MPD when combined with license acquisition information stored in 'pssh' boxes in Ini-  
27 tialization Segments.

28 A **ContentProtection** Descriptor for the mp4 Protection Scheme is used to identify the de-  
29 fault KID, as specified by the 'tenc' box, using the @cenc:default\_KID attribute defined in  
30 [31], section 11.1. The value of the attribute is the KID expressed in UUID string notation.  
31

```
<ContentProtection schemeIdUri="urn:mpeg:dash:mp4protection:2011"  
value="cenc" cenc:default_KID="34e5db32-8625-47cd-ba06-68fca0655a72"/>
```

32  
33 When the default\_KID is present on each Adaptation Set, it allows a player to determine if a  
34 new license needs to be acquired for each Adaptation Set by comparing their default\_KIDs  
35 with each other, and with the default\_KIDs of stored licenses. A player can simply compare  
36 these KID strings and determine what unique licenses are necessary without interpreting license  
37 information specific to each DRM system.

---

### 1 7.6.2.2. ContentProtection Descriptor for UUID Scheme

2 A UUID **ContentProtection** descriptor in the MPD may indicate the availability of a particular DRM scheme for license acquisition. An example is provided below:

```
3 <ContentProtection  
4   schemeIdUri="urn:uuid:xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx"  
   value="DRMNAME version"/>
```

5 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>.

6  
7  
8 This is specified in [8], 5.8.5.2 and is referred to as “**ContentProtection** Descriptor for UUID Scheme” in the following.

### 9 7.6.2.3. Protection System Specific Header Box `cenc:pssh` element in MPD

10  
11 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 [7] and in [31] sec. 11.2.

12 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.

## 13 7.7. Additional Content Protection Constraints

14 The following describes additional constraints for presentations to be conformant with DASH-264/AVC, for both MPD and ISO Media files.

### 15 7.7.1. ISO BMFF Content Protection Constraints

- 16 • 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`.
- 17 • ‘`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

---

1 Representation is sufficient to allow switching between Representations within the Adap-  
2 tation Set without acquiring a new license.

3 Note: ‘pssh’ boxes in Initialization Segments may result in playback failure during  
4 browser playback when a license request is initiated each time an Initialization Segment is  
5 processed, such as the start of each protected Representation, each track selection, and each  
6 bitrate switch. This content requires DASH players that can parse the ‘pssh’ box contents  
7 to determine the duplicate license requests and block them.

8 A `cenc:pssh` element is parsed once per Adaptation Set by a player’s MPD parser, and  
9 duplicate requests are identified by duplicate `cenc:default_KID` values. In this case,  
10 only the DASH player initiates license requests, and may do so per Period, if `cenc:de-`  
11 `fault_KID` is a new value.

## 12 7.7.2. MPD Content Protections Constraints

- 13 • For an encrypted Adaptation Set, **ContentProtection** Descriptors SHALL always be  
14 present in the **AdaptationSet** element, and apply to all contained Representations.
- 15 • A **ContentProtection** Descriptor for the mp4 Protection Scheme with the  
16 `@schemeIdUri` value of "urn:mpeg:dash:mp4protection:2011" and  
17 `@value='cenc'` SHALL be present in the **AdaptationSet** element if the contained  
18 Representations are encrypted.

19 *Note that this allows clients to recognize the Adaptation Set is encrypted with common*  
20 *encryption scheme without the need to understand any system specific UUID descriptors.*

21 The **ContentProtection** Descriptor for the mp4protection scheme SHOULD contain  
22 the optional attribute `@cenc:default_KID`. The ‘tenc’ box that specifies the en-  
23 coded track encryption parameters SHALL be considered the source of truth for the default  
24 key ID value since it contains the `default_KID` field, and is present in the movie box,  
25 as specified in [31], section 8.2.1. The MPD `cenc:default_KID` attribute SHALL match  
26 the ‘tenc’ `default_KID`.

27 *Note that this allows clients to identify the default KID from the MPD using a standard*  
28 *location and format, and makes it accessible to general purpose clients that don’t under-*  
29 *stand the system specific information formats of all DRM schemes that might be signaled.*

- 30 • The `cenc:pssh` element SHOULD be present in the **ContentProtection** De-  
31 scriptor for each UUID Scheme. The base64 encoded contents of the element SHALL be  
32 equivalent to a ‘pssh’ box including its header. The information in the ‘pssh’ box  
33 SHOULD be sufficient to allow for license acquisition.

34 *Note: A player such as DASH.js hosted by a browser may pass the contents of this element*  
35 *through the Encrypted Media Extension (EME) API to the DRM system Content Decryp-*  
36 *tion Module (CDM) with a SystemID equal to the Descriptor’s UUID. This allows clients*  
37 *to acquire a license using only information in the MPD, prior to downloading Segments.*

38 Below is an example of the recommended format for a hypothetical acme DRM service:

```
39  
40 <ContentProtection schemeIdUri="urn:uuid:d0ee2730-09b5-459f-8452-  
41 200e52b37567"  
42 value="Acme DRM 2.0">  
43 <!-- base64 encoded 'pssh' box with SystemID matching the containing  
44 ContentProtection Descriptor -->  
45 <cenc:pssh>  
46 YmFzZTY0IGVuY29kZWQgY29udGVudHMgb2YgkXB
```

---

```
1          zc2iSIGJveCB3aXRoIHRoaxMgU3lzdGVtSUQ=  
2          </cenc:pssh>  
3      </ContentProtection>
```

- 4 • The @value attribute of the **ContentProtection** Descriptor for UUID Scheme  
5 SHOULD contain the DRM system and version in a human readable form.

### 6 **7.7.3. Other Content Protections Constraints**

7 In the case where the ‘pssh’ box element is present in the MPD and in the Initialization Segment,  
8 the ‘pssh’ box element in the MPD SHALL take precedence, because the parameters in the MPD  
9 will be processed first, are easier to update, and can be assumed to be up to date at the time the  
10 MPD is fetched.

11 Recommended scheduling of License and key delivery:

- 12 • Request licenses on initial processing of an MPD if **ContentProtection** Descriptors  
13 or Initialization Segments are available with license acquisition information. This is in-  
14 tended to avoid a large number of synchronized license requests at **MPD@availabil-**  
15 **ityStartTime**.
- 16 • Prefetch licenses for a new Period in advance of its presentation time to allow license  
17 download and processing time, and prevent interruption of continuous decryption and play-  
18 back. Advanced requests will also help prevent a large number of synchronized license  
19 requests during a live presentation at **Period@start** time.

### 20 **7.7.4. Additional Constraints for Periodic Re-Authorization**

- 21 • Key rotation should not occur within individual segments, as their duration is typically short  
22 enough to enable the intended use cases.
- 23 • Each Movie Fragment SHOULD contain one ‘pssh’ in each ‘moov’ box per **System-**  
24 **temID** that contains sufficient information for the DRM system with matching **System-**  
25 **temID** to obtain protected keys for this movie fragment, when combined with:
  - 26 ○ Information from ‘pssh’ in ‘moov’ or **cenc:pssh** in MPD.
  - 27 ○ KID associated with each sample from ‘seig’ sample group description box.
  - 28 ○ Sample to group boxes that list all the samples that use a particular KID.
- 29 • The KID should be observable by the player by reading the clear **key\_ids** in PSSH defini-  
30 tion v1.
- 31 • If the key is does not need to be retrieved, a pssh update may not result in a license request.
- 32 • If **key\_id** cannot be observed, the player may perform binary comparison of pssh seg-  
33 ments to understand updates.

### 34 **7.7.5. Encryption of Different Representations**

35 Representations contained in one Adaptation Set SHALL be protected by the same license for each  
36 protection system (“DRM”), and SHALL have the same value of ‘**default\_KID**’ in their  
37 ‘**tenc**’ boxes in their Initialization Segments. This is to enable seamless switching within Adap-  
38 tation Sets, which is generally not possible if a new DRM license needs to be authorized, client  
39 bound, generated, downloaded, and processed for each new Representation.

40 In the case of key rotation, if root licenses are used, the same requirement applies to the root li-  
41 censes (one license per Adaptation Set for each DRM), and also means all Representations SHALL

---

1 have the same value of ‘default\_KID’ in their ‘tenc’ boxes in their Initialization Segments.  
2 The use of root and leaf licenses is optional and DRM specific, but leaf licenses are typically  
3 delivered in band to allow real time license acquisition, and do not require repeating client authen-  
4 tication, authorization, and rebuilding the security context with each key change in order to enable  
5 continuous playback without interruption cause by key acquisition or license processing.  
6 In cases where SD and HD and UHD Representations are contained in one presentation, different  
7 license rights may be required for each quality level and may be sold separately. If different li-  
8 censes are required for different quality levels, then it is necessary to create separate Adaptation  
9 Sets for each quality level, each with a different license and value of ‘default\_KID’.  
10 Representations that are equivalent resolution and bitrate but encrypted with different keys may  
11 be included in different Adaptation Sets. Seamless switching between UHD, HD and SD Re-  
12 presentations is difficult because these quality levels typically use different decryption licenses and  
13 keys, use different DRM output rules (prohibit analog interfaces, require resolution down-scaling,  
14 require HDCP encryption on output, etc.), and use different decoding parameters for e.g. subsam-  
15 pling, codec, profile, bit depth, aspect ratios and color spaces.  
16 If any Representation is encrypted in an Adaptation Set, then all must be encrypted using the same  
17 default\_KID in the Track Encryption Box (‘tenc’) to avoid realtime changes to the DRM  
18 licenses and security context. KID values may change over time (“key rotation”) as specified in  
19 Common Encryption and a particular DRM system.  
20 For all Representations within an Adaptation Set with @bitstreamSwitching=”false” (de-  
21 fault), the following parameters shall apply.  
22 • ‘tenc’ default\_KID shall be equal for all Representations

### 23 **7.7.6. Encryption of Multiple Periods**

24 Periods SHALL only change default\_KID and corresponding license at the start of a new Pe-  
25 riod corresponding to a different file. A different file is indicated by a different default\_KID  
26 signaled in the ‘tenc’ box in the Initialization Segment.  
27 A file associated with a single license may be continued over multiple Periods by being referenced  
28 by multiple Representations over multiple Periods (for instance, a program interspersed with ad  
29 Periods). A client can recognize the same cenc:default\_KID value and avoid having to  
30 download the same license again; but the DRM system may require a complete erase and rebuild  
31 of the security context, including all key material, samples in process, etc., between Periods with  
32 different licenses or no license (between protected and clear Periods).

### 33 **7.7.7. DRM System Identification**

34 The DRM system is signaled in the MPD and ‘pssh’ boxes with a SystemID. A list of known  
35 DRMs can be found in the DASH identifier repository available here:  
36 <http://www.dashif.org/identifiers/protection>.

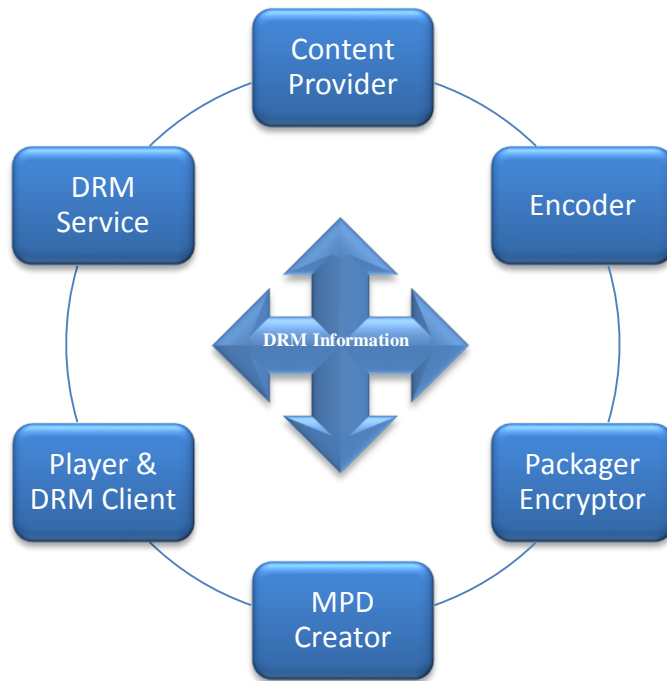
### 37 **7.7.8. Protection of Media Presentations that Include SD, HD and UHD Adap-** 38 **tation Sets**

39 Per DASH IF interop points, Representations with separate keys, licenses, and license policy are  
40 contained in different Adaptation Sets.  
41 Adaptive bitrate switching can function automatically within an Adaptation Set without changing  
42 keys, licenses, robustness and output rules, etc.

---

1 A player may download licenses for multiple Adaptation Sets in a Group, and seamlessly switch  
2 between them if it is able. Seamless switching between Adaptation Sets is allowed, but not re-  
3 quired. DASH may need to signal which Adaptation Sets are intended for seamless switching, i.e.  
4 have identical source content, same picture aspect ratio, same exact rescaled pixel registration,  
5 same sample description (e.g. 'avc3'), same initialization behavior (@bitstreamSwitch-  
6 ing=true/false), same Timescale and @timescale, and are mutually time-aligned.  
7 The DASH-IF interop points are intended to make bitrate switching within an Adaptation Set sim-  
8 ple and automatic, whether Representations are encrypted or not. Placement of Representations  
9 in different Adaptation Sets informs players that those Representations need to be initialized with  
10 different parameters, such as a different key and license. The full initialization process is repeated  
11 per Period. Adaptation Sets with @bitstreamSwitching = "true" only need to be initial-  
12 ized once per Period. Adaptation Sets with @bitstreamSwitching = "false" need to be  
13 partially re-initialized on each Representation switch (to change the SPS parameter sets referenced  
14 from NALs to those stored in in the containing track's 'avcC'), but most initialized parameters  
15 such as timescale, codec Profile/Level, display buffer size, colorspace, etc.; and licenses and the  
16 DRM system ... do not need to be changed.  
17 Fetching and resetting keys and licenses during adaptive switching requires processing Initializa-  
18 tion Segments with different 'tenc' default\_KID and possibly 'pssh' boxes. That may not be  
19 seamless, especially in browser playback where the decoders are only aware of player switching  
20 when an Initialization Segment flows through the MSE buffer and a needKey() event is raised  
21 via EME.  
22 Note that switching between Adaptation Sets with different Media Profiles could be restricted by  
23 key and license policy, e.g. the user only purchased SD rights, the player only has analog output  
24 and HD content requires a protected digital output, UHD content requires hardware protected  
25 DRM, etc.  
26 Implementations that seamlessly switch between Representations with different keys and policies  
27 generally require a standardized presentation ID or content ID system that associates multiple keys  
28 and licenses to that ID and presentation, then downloads only the keys/licenses authorized for that  
29 user and device (e.g. SD or HD+SD). The player must then install those licenses and use player  
30 logic to select only Representations in an Adaptation Set for which a license is installed and output  
31 controls, display configuration, etc. allow playback (e.g. only Representations keyed for an in-  
32 stalled SD license). Players and license servers without this pre-configuration protocol and adap-  
33 tive switching logic will encounter key/license requests in the process of adaptive switching, and  
34 may find output blocked by different license policies, user rights, etc.

## 1 7.8. Workflow Overview



2  
3 **Figure 20 Logical Roles that Exchange DRM Information and Media**

4 Figure 20 shows logical entities that may send or receive DRM information such as media keys,  
5 asset identifiers, licenses, and license acquisition information. A physical entity may combine  
6 multiple logical roles, and the point of origin for information, such as media keys and asset identifiers,  
7 can differ; so various information flows are possible. This is an informative example of  
8 how the roles are distributed to facilitate the description of workflow and use cases. Alternative  
9 roles and functions can be applied to create conformant content.

### 10 **Description of Logical Roles:**

11 **Content Provider** – A publisher who provides the rights and rules for delivering protected media,  
12 also possibly source media (mezzanine format, for transcoding), asset identifiers, key identifiers  
13 (KID), key values, encoding instructions, and content description metadata.

14 **Encoder** – A service provider who encodes Adaptation Sets in a specified media format, number  
15 of streams, range of bitrates and resolutions, seamless switching constraints, etc., possibly determined  
16 by the publisher. An asset identifier needs to be assigned to each encoded track in order to  
17 associate a key identifier, a Representation element in an MPD, a possible ‘pssh’ box in the file  
18 header, and a DRM license separately downloaded.

19 **Packager / Encryptor** – A service provider who encrypts and packages media files, inserting  
20 default\_KID in the file header ‘tenc’ box, initialization vectors and subsample byte ranges  
21 in track fragments indexed by ‘saio’ and ‘saiz’ boxes, and possibly packages ‘pssh’ boxes  
22 containing license acquisition information (from the DRM Provider) in the file header. Tracks that  
23 are partially encrypted or encrypted with multiple keys require sample to group boxes and sample  
24 group description boxes in each track fragment to associate different KIDs to groups of samples.  
25 The Packager could originate values for KIDs, media keys, encryption layout, etc., then send that

---

1 information to other entities that need it, including the DRM Provider and Streamer, and probably  
2 the Content Provider. However, the Packager could receive that information from a different point  
3 of origin, such as the Content Provider or DRM Provider.

4 **MPD Creator** – The MPD Creator is assumed to create one or more types of DASH MPD, and  
5 provide indexing of Segments and/or ‘`sidx`’ indexes for download so that players can byte range  
6 index Subsegments. The MPD must include descriptors for Common Encryption and DRM key  
7 management systems, and SHOULD include identification of the `default_KID` for each `Ad-`  
8 `aptationSet` element, and sufficient information in `UUID ContentProtection De-`  
9 `scriptor` elements to acquire a DRM license. The `default_KID` is available from the Packager  
10 and any other role that created it, and the DRM specific information is available from the DRM  
11 Provider.

12 **Player / DRM Client** – Gets information from different sources: MPD, Media files and DRM  
13 License.

14 **DRM Service** – The DRM Provider creates licenses containing a protected media key that can  
15 only be decrypted by a trusted client.

16 The DRM Provider needs to know the `default_KID` and DRM `SystemID` and possibly other  
17 information like asset ID and player domain ID in order to create and download one or more li-  
18 censes required for a Presentation on a particular device. Each DRM system has different license  
19 acquisition information, a slightly different license acquisition protocol, and a different license  
20 format with different playback rules, output rules, revocation and renewal system, etc. The DRM  
21 Provider typically must supply the Streamer and the Packager license acquisition information for  
22 each `UUID ContentProtection Descriptor` element or ‘`pssh`’ box, respectively.

23 The DRM Service may also provide logic to manage key rotation, DRM domain management,  
24 revocation and renewal and other content protection related features.

25

26 Figure 21 shows a simple workflow with `pssh` information in the Initialization Segment for in-  
27 formational purpose.



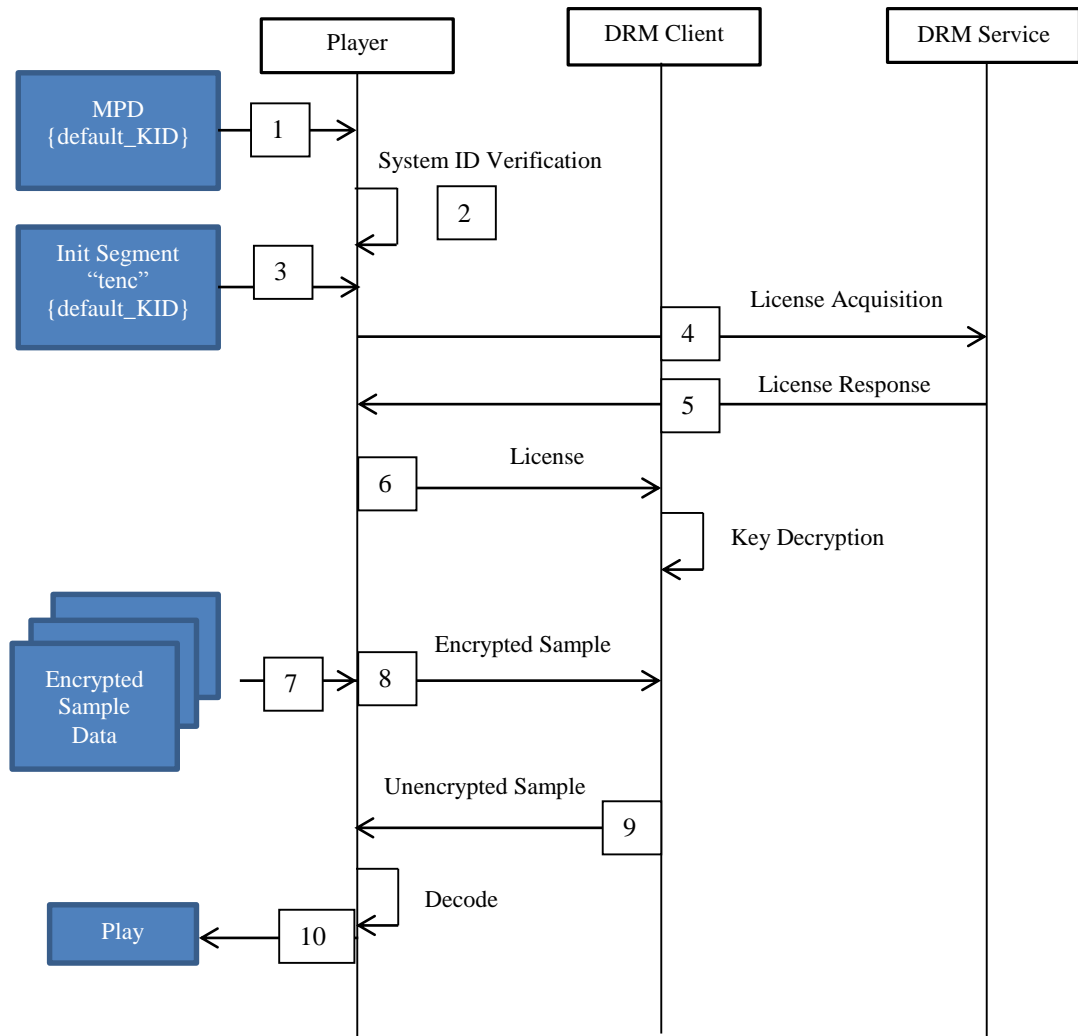


Figure 21 Example of Information flow for DRM license retrieval

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16

[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.

- 
- 1 [4] Decryption Key acquisition can be performed either by the Player or the DRM Client.
- 2 [5] DRM License / Decryption Key response includes the required material for enabling access.
- 3 [6] DRM licenses/rights need not be stored in order to look up a key using KID values stored in
- 4 the file and decrypt media samples using the encryption parameters stored in each track.
- 5 [7] The Player requesting encrypted sample data.
- 6 [8] The Player provides encrypted sample data to the DRM Client for decryption using the decryp-
- 7 tion key. How the DRM system locates the identified decryption key is left to the DRM.
- 8 [9] The Player received unencrypted sample data from the DRM Client.

## 9 **8. DASH-IF Interoperability Points**

### 10 **8.1. Introduction**

11 This version of the document defines Interoperability Points in this section. Earlier versions of this

12 document, especially version 2 [2] defines legacy IOPs.

### 13 **8.2. DASH-AVC/264 main**

#### 14 **8.2.1. Introduction**

15 The scope of the DASH-AVC/264 main interoperability point is basic support of high-quality

16 video distribution over the top based on H.264/AVC up to 1080p. Both, live and on-demand ser-

17 vices are supported.

18 The compliance to DASH-AVC/264 main may be signaled by a @profiles attribute with the

19 value "<http://dashif.org/guidelines/dash264main>"

#### 20 **8.2.2. Definition**

21 A DASH client conforms to the IOP by supporting at least the following features:

- 22 • All DASH-related features as defined in section 3 of this document.
- 23 • The requirements and guidelines in section 4.9.2 for simple live operation.
- 24 • The requirements and guidelines in section 5.6.1 for server-based ad insertion.
- 25 • H.264/MPEG AVC Progressive High Profile at level 4.0 as defined in section 6.2 together
- 26 with all AVC-related requirements and recommendation in section 6.2.
- 27 • MPEG-4 HE-AAC v2 level 2 profile audio codec as defined in section 6.3. Dynamic Range
- 28 Control is not expected to be supported.
- 29 • subtitle and closed captioning support
  - 30 ○ using SMPTE-TT as defined in section 6.4.2
    - 31 ▪ For On-Demand single file download is sufficient.
    - 32 ▪ For live services and/or if key rotation is to be supported, the encapsulation
    - 33 into ISO BMFF is necessary.
  - 34 ○ Using CEA-608/708 as defined in section 6.4.3.
- 35 • content protection based on common encryption and key rotation as defined in section 7.
- 36 And specifically, the client supports MPD-based parsing and movie box based parsing of
- 37 DRM related parameters for common encryption.

---

1 Content shall only be authored claiming conformance to this IOP if such a client can properly play  
2 the content. In addition, the content shall follow the mandatory aspects and should take into ac-  
3 count the recommendations and guidelines for content authoring documented in section 3 (DASH  
4 features), section 4.9.2 (simple live operation), section 5.6.1 (server-based ad insertion), AVC-  
5 related issues in section 6.2, section 6.3 (audio), section 6.4.2 (SMPTE-TT), section 6.4.3 (CEA-  
6 608/708), and section 7 (Content Protection).

7 If content is offered claiming conformance to this IOP, the content author is encouraged to use the  
8 HTTP-URL construction as defined in [7], section 5.1.4.

## 9 **8.3. DASH-AVC/264 high**

### 10 **8.3.1. Introduction**

11 The scope of the DASH-AVC/264 interoperability point is support of high-quality video distribu-  
12 tion over the top based on H.264/AVC up to 1080p. Both, live and on-demand services are sup-  
13 ported as well as features for main live and advanced ad insertion.

14 The compliance to DASH-AVC/264 may be signaled by a @profiles attribute with the value  
15 "http://dashif.org/guidelines/dash264high"

### 16 **8.3.2. Definition**

17 A client that attempts to consume content generated conforming to this IOP shall support the fol-  
18 lowing features:

- 19 • All features required for DASH-264/AVC main as defined in section 8.2.
- 20 • The client requirements and recommendations for the main live operation as defined in  
21 section 4.9.3.

22 Content shall only be authored claiming conformance to this IOP if such a client can properly play  
23 the content. In addition, the content shall follow the mandatory aspects and should take into ac-  
24 count the recommendations and guidelines for content authoring documented in section 8.2  
25 (DASH-264/AVC main), section 4.9.3 (main live operation), and section 5.6.2 (app-based ad in-  
26 sertation).

27 If content is offered claiming conformance to this IOP, the content author is encouraged to use the  
28 HTTP-URL construction as defined in [7], section 5.1.4.

## 29 **8.4. DASH-IF IOP simple**

### 30 **8.4.1. Introduction**

31 The scope of the DASH-IF IOP simple interoperability point is the basic support of efficient high-  
32 quality video distribution over the top with HD video up to 1080p including support for HEVC 8  
33 bit.

34 The compliance to *DASH-IF IOP simple* may be signaled by a @profiles attribute with the  
35 value "http://dashif.org/guidelines/dash-if-simple"

### 36 **8.4.2. Definition**

37 A DASH client conforms to the IOP by supporting at least the following features:

- 38 • All DASH-related features as defined in section 3 of this document.
- 39 • The requirements and guidelines in section 4.9.2 for simple live operation.
- 40 • The requirements and guidelines in section 5.6.1 for server-based ad insertion.

- 
- 1 • H.264/MPEG AVC Progressive High Profile at level 4.0 as defined in section 6.2 together  
2 with all AVC-related requirements and recommendation in section 6.2.
  - 3 • H.265/MPEG-H HEVC Main Profile Main Tier at level 4.1 as defined in section 6.2 to-  
4 gether with all HEVC-related requirements and recommendation in section 6.2.
  - 5 • MPEG-4 HE-AAC v2 level 2 profile audio codec as defined in section 6.3. Dynamic Range  
6 Control is not expected to be supported.
  - 7 • subtitle and closed captioning support  
8 ○ using SMPTE-TT as defined in section 6.4.2  
9     ▪ For On-Demand single file download is sufficient.  
10     ▪ For live services and/or if key rotation is to be supported, the encapsulation  
11 into ISO BMFF is necessary.  
12 ○ Using CEA-608/708 as defined in section 6.4.3.
  - 13 • content protection based on common encryption and key rotation as defined in section 7.  
14 And specifically, the client supports MPD-based parsing and movie box based parsing of  
15 DRM related parameters for common encryption.

16 Content shall only be authored claiming conformance to this IOP if such a client can properly play  
17 the content. In addition, the content shall follow the mandatory aspects and should take into ac-  
18 count the recommendations and guidelines for content authoring documented in section 3 (DASH  
19 features), section 4.9.2 (simple live operation), section 5.6.1 (server-based ad insertion), section  
20 6.2 (video), section 6.3 (audio), section 6.4.2 (SMPTE-TT), section 6.4.3 (CEA-608/708), and  
21 section 7 (Content Protection).

22 If content is offered claiming conformance to this IOP, the content author is encouraged to use the  
23 HTTP-URL construction as defined in [7], section 5.1.4.

## 24 **8.5. DASH-IF IOP Main**

### 25 **8.5.1. Introduction**

26 For the support of broad set of use cases the DASH-IF IOP Main Interoperability Point is defined.  
27 In addition the features of DASH-264/AVC main as defined in section 8.2 the interoperability  
28 point requires DASH clients for real-time segment parsing and 10-bit HEVC.

29 The compliance to *DASH-IF IOP main* may be signalled by a `@profile` attribute with the value  
30 "`http://dashif.org/guidelines/dash-if-main`"

### 31 **8.5.2. Definition**

32 A client that attempts to consume content generated conforming to this IOP shall support the fol-  
33 lowing features:

- 34 • All features required for DASH-264/AVC high as defined in section 8.3.
- 35 • H.265/MPEG-H HEVC Main Profile Main Tier at level 4.1 as defined in section 6.2 to-  
36 gether with all HEVC-related requirements and recommendation in section 6.2.
- 37 • H.265/MPEG-H HEVC Main 10 Profile Main Tier at level 4.1 as defined in section 6.2  
38 together with all HEVC-related requirements and recommendation in section 6.2.

39 Content shall only be authored claiming conformance to this IOP if such a client can properly play  
40 the content. In addition, the content shall follow the mandatory aspects and should take into ac-  
41 count the recommendations and guidelines for content authoring documented in section 8.3 and  
42 HEVC-related issues in section 6.2.

1 If the content is authored such that it also conforms to DASH-264/AVC high as defined in section  
 2 8.3, then the profile identifier for DASH-264/AVC high shall be added as well. If the profile identifier  
 3 is missing, the content may be considered as HEVC only content.  
 4 If content is offered claiming conformance to this IOP, the content author is encouraged to use the  
 5 HTTP-URL construction as defined in [7], section 5.1.4.

## 6 **9. Multi-Channel Audio Extension**

### 7 **9.1. Scope**

8 The Scope of the Multichannel Audio Extension is the support of audio with additional channels  
 9 and codecs beyond the basic audio support as specified in the DASH-AVC/264 base, which is  
 10 limited to Stereo HE-AAC. Multichannel audio is widely supported in all distribution channels  
 11 today, including broadcast, optical disc, and digital delivery of audio, including wide support in  
 12 adaptive streaming delivery.

13 It is expected that clients may choose which formats (codecs) they support.

### 14 **9.2. Technologies**

#### 15 **9.2.1. Dolby Multichannel Technologies**

##### 16 **9.2.1.1. Overview**

17 The considered technologies from Dolby for advanced audio support are:

- 18 • Enhanced AC-3 (Dolby Digital Plus) [36]
- 19 • Dolby TrueHD [37]
- 20 • AC-4 [64]

##### 21 **9.2.1.2. DASH-specific issues**

22 In the context of DASH, the following applies:

- 23 • The signaling of the different audio codecs for the codecs parameters is documented in  
 24 [36], [37] and [64] which also provides information on ISO BMFF encapsulation.
- 25 • For E-AC-3 and AC-4 the Audio Channel Configuration shall use the "  
 26 tag:dolby.com,2014:dash:audio\_channel\_configuration:2011" as  
 27 defined at <http://dashif.org/identifiers/audio-source-data/>.

28 **Table 21 Dolby Technologies: Codec Parameters and ISO BMFF encapsulation**

Codec	Codec Parameter	ISO BMFF Encapsulation	SAP type
<b>Enhanced AC-3 [36]</b>	ec-3	ETSI TS 102 366 Annex F [36]	1
<b>Dolby TrueHD</b>	m1pa	Dolby [37]	1
<b>AC-4</b>	ac-4	ETSI TS 103 190-1 Annex E [64]	1

#### 29 **9.2.2. DTS-HD**

##### 30 **9.2.2.1. Overview**

31 DTS-HD [38] comprises a number of profiles optimized for specific applications. More information  
 32 about DTS-HD and the DTS-HD profiles can be found at [www.dts.com](http://www.dts.com).

1 **9.2.2.2. DASH-specific issues**

2 For all DTS formats SAP is always 1.

3 The signaling of the various DTS-HD profiles is documented in DTS 9302J81100 [35]. DTS  
4 9302J81100 [35] also provides information on ISO BMFF encapsulation.

5 Additional information on constraints for seamless switching and signaling DTS audio tracks in  
6 the MPD is described in DTS specification 9302K62400 [40].

7 **Table 22: DTS Codec Parameters and ISO BMFF encapsulation**

Codec	Codec parameter	Pa-	ISO BMFF Encapsu- lation	SAP type
<b>DTS Digital Surround</b>	dtsc		DTS 9302J81100 [35]	1
<b>DTS-HD High Resolution and DTS-HD Master Audio</b>	dtsh			
<b>DTS Express</b>	dtse			
<b>DTS-HD Lossless (no core)</b>	dtsl			

8

9 **9.2.3. MPEG Surround**

10 **9.2.3.1. Overview**

11 MPEG Surround, as defined in ISO/IEC 23003-1:2007 [39], is a scheme for coding multichannel  
12 signals based on a down-mixed signal of the original multichannel signal, and associated spatial  
13 parameters. The down-mix shall be coded with MPEG-4 High Efficiency AAC v2 according to  
14 section 5.3.3.

15 MPEG Surround shall comply with level 4 of the Baseline MPEG Surround profile.

16 **9.2.3.2. DASH-specific issues**

17 In the context of DASH, the following applies for audio codecs

- 18 • The signaling of the different audio codecs for the codecs parameters is according to  
19 RFC6381 [11] is documented in Table 23. Table 23 also provides information on ISO  
20 BMFF encapsulation.
- 21 • The content is expected to be prepared according to the MPEG-DASH Implementation  
22 Guidelines [7] to make sure each (sub-)segment starts with a SAP of type 1.

23 **Table 23 Codecs parameter according to RFC6381 [11] and ISO BMFF encapsulation for MPEG Surround**  
24 **codec**

Codec	Codec Parameter	ISO BMFF Encapsulation	SAP type
<b>MPEG Surround [39]</b>	mp4a.40.30	ISO/IEC 14496-14 [8]	1

25 **Note:** Since MPEG Surround is based on a down-mix coded with AAC-LC and HE-AAC, for the  
26 above mentioned “Codec Parameters” the following is implied:

27 mp4a.40.30 = AOT 2 + AOT 5 + AOT 30

---

## 1 9.2.4. MPEG-4 High Efficiency AAC Profile v2, level 6

### 2 9.2.4.1. Overview

3 Support for multichannel content is available in the HE-AACv2 Profile, starting with level 4 for  
4 5.1 and level 6 for 7.1. All MPEG-4 HE-AAC multichannel profiles are fully compatible with the  
5 DASH-AVC/264 baseline interoperability point for stereo audio, i.e. all multichannel decoders  
6 can decode DASH-IF IOPS stereo content.

### 7 9.2.4.2. DASH-specific issues

8 In the context of DASH, the following applies for the High Efficiency AAC v2 Profile

- 9 • The content shall be prepared according to the MPEG-DASH Implementation Guidelines  
10 [7] to make sure each (sub-)segment starts with a SAP of type 1.
- 11 • Signaling of profile levels is not supported in RFC 6381 but the channel configuration shall  
12 be signaled by means of the **ChannelConfiguration** element in the MPD.
- 13 • The signaling of MPEG-4 High Efficiency AAC v2 for the codecs parameters is according  
14 to RFC6381 [11] and is documented in Table 24. Table 24 also provides information on  
15 the ISO BMFF encapsulation.
- 16 • For all HE-AAC bitstreams, explicit backward-compatible signaling of SBR shall be used.
- 17 • The content should be prepared incorporating loudness and dynamic range information into  
18 the bitstream also considering DRC Presentation Mode in ISO/IEC 14496-3 [12], Amd. 4.
- 19 • Decoders shall support decoding of loudness and dynamic range related information, i.e.  
20 dynamic\_range\_info() and MPEG4\_ancillary\_data() in the bitstream.

21 **Table 24 Codecs parameter according to RFC6381 [11] and ISO BMFF encapsulation**

Codec	Codec Parameter	ISO BMFF Encapsulation	SAP type
MPEG-4 AAC Profile [12]	mp4a.40.2	ISO/IEC 14496-14 [13]	1
MPEG-4 HE-AAC Profile [12]	mp4a.40.5	ISO/IEC 14496-14 [13]	1
MPEG-4 HE-AAC v2 Profile [12]	mp4a.40.29	ISO/IEC 14496-14 [13]	1

22 **Note:** Since both, HE-AAC and HE-AACv2 are based on AAC-LC, for the above mentioned “Co-  
23 dec Parameters” the following is implied:

24 mp4a.40.5 = AOT 2 + AOT 5

## 25 9.2.5. MPEG-H 3D Audio

### 26 9.2.5.1. Overview

27 MPEG-H 3D Audio (MHA) including carriage in the ISO BMFF is specified in Amd. 2 of ISO/IEC  
28 23008-3 [65].

### 29 9.2.5.2. DASH-specific issues

30 Storage of MHA elementary streams in the ISO BMFF shall be according to ISO/IEC 23008-3  
31 [65] with the following constraints:

- 32 • An audio sample shall consist of a single MHA access unit
- 33 • The parameter values of MHADecoderConfigurationRecord and MHASam-  
34 pleEntry shall be consistent with the configuration in the MHA elementary stream

- `referenceChannelLayout` shall be 'ChannelConfiguration' according to ISO/IEC 23001-8 [50].
- MHA bit streams shall contain an extension element of type `ID_EXT_ELE_AUDIOPRE-ROLL` as specified in ISO/IEC 23008-3 sub-clause 5.5.5. [65]. 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 25** Codecs parameter and ISO BMFF encapsulation

Code	Codec Parameter	ISO BMFF Encapsulation	SAP Type
MPEG-H 3D Audio	<code>mha [1, 2]</code>	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 seamlessly 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 [7], section 5.1.4.



---

## 1 **9.4.2. Dolby Extensions**

### 2 **9.4.2.1. Introduction**

3 For the support of Dolby advanced audio support, three additional extensions are defined.  
4 Conformance to *DASH-IF multichannel audio extension with Enhanced AC-3* (Dolby Digital Plus)  
5 [36] may be signaled by an @profile attribute with the value  
6 "http://dashif.org/guidelines/dashif#ec-3".

7 Conformance to *DASH-IF multichannel extension with Dolby TrueHD* may be signaled by an  
8 @profile attribute with the value "http://dashif.org/guide-  
9 lines/dashif#mlpa".

10 Conformance to *DASH-IF multichannel extension with AC-4* may be signaled by an @profile  
11 attribute with the value "http://dashif.org/guidelines/dashif#ac-4".

### 12 **9.4.2.2. Supporters**

13 These extensions are supported by the following DASH IF members: Dolby, DTS, Fraunhofer,  
14 BuyDRM, Sony.

### 15 **9.4.2.3. Definition**

16 Content may be authored claiming conformance to *DASH-IF multichannel audio extension with*  
17 *Enhanced AC-3*

- 18 • if the content is multichannel audio content as defined in section 9.4.1, and
- 19 • if a client can properly play the content by supporting at least the following features
  - 20 • all multichannel audio client features as defined in section 9.4.1
  - 21 • Enhanced AC-3 (Dolby Digital Plus) [36] and the DASH-specific features defined in
  - 22 section 9.2.1.2

23 Content may be authored claiming conformance to *DASH-IF multichannel extension with Dolby*  
24 *TrueHD*

- 25 • if the content is multichannel audio content as defined in section 9.4.1, and
- 26 • if a client can be properly play the content by supporting at least the following features
  - 27 • all multichannel audio client features as defined in section 9.4.1
  - 28 • Dolby TrueHD and the DASH-specific features defined in section 9.2.1.2

29 Content may be authored claiming conformance to *DASH-IF multichannel extension with AC-4*  
30 *DASH-IF multichannel extension with Dolby TrueHD*

- 31 • if the content is multichannel audio content as defined in section 9.4.1, and
- 32 • if a client can be properly play the content by supporting at least the following features
  - 33 • all multichannel audio client features as defined in section 9.4.1
  - 34 • AC-4 and the DASH-specific features defined in section 9.2.1.2

## 36 **9.4.3. DTS-HD Interoperability Points**

### 37 **9.4.3.1. Introduction**

38 For the support of DTS advanced audio support, four additional extensions are defined.  
39 Conformance to *DASH-IF multichannel audio extension with DTS Digital Surround* may be sig-  
40 naled by a @profile attribute with value "http://dashif.org/guide-  
41 lines/dashif#dtsc".

---

1 Conformance to *DASH-IF multichannel audio extension with DTS-HD High Resolution and DTS-*  
2 *HD Master Audio* may be signaled by a @profile attribute with value  
3 "http://dashif.org/guidelines/dashif#dtsh"  
4 Conformance to *DASH-IF multichannel audio extension with DTS Express* may be signaled by a  
5 @profile attribute with value "http://dashif.org/guidelines/dashif#dtse"  
6 Conformance to *DASH-IF multichannel extension with DTS-HD Lossless (no core)* may be sig-  
7 naled by a @profile attribute with value "http://dashif.org/guide-  
8 lines/dashif#dtsl"

#### 9 **9.4.3.2. Supporters**

10 These extensions are supported by the following DASH IF members: Dolby, DTS, Fraunhofer,  
11 BuyDRM, Sony.

#### 12 **9.4.3.3. Definition**

13 Content may be authored claiming conformance to *DASH-IF multichannel audio extension with*  
14 *DTS Digital Surround*

- 15 • if the content is multichannel audio content as defined in section 9.4.1, and
- 16 • if a client can be properly play the content by supporting at least the following features
  - 17 • all multichannel audio client features as defined in section 9.4.1
  - 18 • DTS and the DASH-specific features defined in section 9.2.2.2

19 Content may be authored claiming conformance to *DASH-IF multichannel audio extension with*  
20 *DTS-HD High Resolution and DTS-HD Master Audio*

- 21 • if the content is multichannel audio content as defined in section 9.4.1, and
- 22 • if a client can be properly play the content by supporting at least the following features
  - 23 • all multichannel audio client features as defined in section 9.4.1
  - 24 • DTS-HD High Resolution and DTS-HD Master Audio and the DASH-specific features
  - 25 defined in section 9.2.2.2

26 Content may be authored claiming conformance to *DASH-IF multichannel audio extension with*  
27 *DTS Express*

- 28 • if the content is multichannel audio content as defined in section 9.4.1, and
- 29 • if a client can be properly play the content by supporting at least the following features
  - 30 • all multichannel audio client features as defined in section 9.4.1
  - 31 • DTS-HD Express and the DASH-specific features defined in section 9.2.2.2

32 Content may be authored claiming conformance to *DASH-IF multichannel extension with DTS-*  
33 *HD Lossless (no core)*

- 34 • if the content is multichannel audio content as defined in section 9.4.1, and
- 35 • if a client can be properly play the content by supporting at least the following features
  - 36 • all multichannel audio client features as defined in section 9.4.1
  - 37 • DTS-HD Lossless (no core) and the DASH-specific features defined in section 9.2.2.2

### 38 **9.4.4. MPEG Surround Interoperability Points**

#### 39 **9.4.4.1. Introduction**

40 For the support of MPEG Surround advanced audio support the following extension is defined.

---

1 Conformance to *DASH-IF multichannel audio extension with MPEG Surround* according to  
2 ISO/IEC 23003-1:2007 [39] may be signaled by an @profile attribute with the value  
3 "http://dashif.org/guidelines/dashif#mps".

#### 4 **9.4.4.2. Supporters**

5 These extensions are supported by the following DASH IF members: Dolby, DTS, Fraunhofer,  
6 BuyDRM, Sony.

#### 7 **9.4.4.3. Definition**

8 Content may be authored claiming conformance to *DASH-IF multichannel audio extension with*  
9 *MPEG Surround*

- 10 • if the content is multichannel audio content as defined in section 9.4.1, and
- 11 • if a client can be properly play the content by supporting at least the following features
  - 12 • all multichannel audio client features as defined in section 9.4.1
  - 13 • ISO/IEC 23003-1:2007 and the DASH-specific features defined in section 9.2.3.2

### 14 **9.4.5. MPEG HE-AAC Multichannel Interoperability Points**

#### 15 **9.4.5.1. Introduction**

16 Conformance to *DASH-IF multichannel audio extension with HE-AACv2 level 4* [12] may be sig-  
17 naled by an @profile attribute with the value "http://dashif.org/guide-  
18 lines/dashif#heaac-mc51".

19 Conformance to *DASH-IF multichannel audio extension with HE-AACv2 level 6* [12] may be sig-  
20 naled by an @profile attribute with the value "http://dashif.org/guide-  
21 lines/dashif#heaac-mc71".

#### 22 **9.4.5.2. Supporters**

23 These extensions are supported by the following DASH IF members: Dolby, DTS, Fraunhofer,  
24 BuyDRM, Sony.

#### 25 **9.4.5.3. Definition**

26 Content may be authored claiming conformance to *DASH-IF multichannel audio extension with*  
27 *HE-AACv2 level 4*

- 28 • if the content is multichannel audio content as defined in section 9.4.1, and
- 29 • if a client can be properly play the content by supporting at least the following features
  - 30 • all multichannel audio client features as defined in section 9.4.1
  - 31 • HE-AACv2 level 4 [12] and the DASH-specific features defined in section 9.2.4.2

32 Content may be authored claiming conformance to *DASH-IF multichannel audio extension with*  
33 *HE-AACv2 level 6*

- 34 • if the content is multichannel audio content as defined in section 9.4.1, and
- 35 • if a client can be properly play the content by supporting at least the following features
  - 36 • all multichannel audio client features as defined in section 9.4.1
  - 37 • HE-AACv2 level 6 [12] and the DASH-specific features defined in section 9.2.4.2

---

## 1 **9.4.6. MPEG-H 3D Audio Interoperability Points**

### 2 **9.4.6.1. Introduction**

3 Compliance to DASH-IF multichannel audio extension with MPEG-H 3D Audio [x] may be sig-  
4 naled by a @profile attribute with the value `http://dashif.org/guide-`  
5 `lines/dashif#mha[1,2]`.

### 6 **9.4.6.3. Definition**

7 Content may be authored claiming conformance to DASH-IF multichannel audio extension with  
8 MPEG-H 3D Audio

- 9 • if the content is multichannel audio content as defined in section 9.4.1, and
- 10 • if a client can properly play the content by supporting at least the following features:
  - 11 ○ all multichannel audio client features as defined in section 9.4.1,
  - 12 ○ MHA and the DASH-specific features defined in section 9.4.6.

# 1 **Annex A Examples for Profile Signalling**

## 2 **Example 1**

3 In this case DASH-IF IOP content is offered, but in addition a non-conforming Adaptation Set  
4 is added.

5 Here is an example for an MPD:

- 6 • **MPD**@profiles="urn:mpeg:dash:profile:isoff-on-demand:2011,  
7 http://dashif.org/guidelines/dash264"
  - 8 ○ **AdaptationSet**@profiles="urn:mpeg:dash:profile:isoff-on-de-  
9 mand:2011, http://dashif.org/guidelines/dash264"
  - 10 ○ **AdaptationSet**@profiles="http://dashif.org/guidelines/dash264"
  - 11 ○ **AdaptationSet**@profiles="urn:mpeg:dash:profile:isoff-on-de-  
12 mand:2011"

13 Pruning process for IOP <http://dashif.org/guidelines/dash264> results in

- 14 • **MPD**@profiles="http://dashif.org/guidelines/dash264"
  - 15 ○ **AdaptationSet**@profiles="http://dashif.org/guidelines/dash264"
  - 16 ○ **AdaptationSet**@profiles="http://dashif.org/guidelines/dash264"

17 It is now required that the pruned MPD conforms to DASH-IF IOP.

## 18 **Example 2**

19 In this case DASH-IF IOP content is offered, but in addition a non-conforming Adaptation Set  
20 is added and one DASH-IF Example Extension Adaptation Set is added with the virtual IOP  
21 signal <http://dashif.org/guidelines/dashif#extension-example>.

22 Here is an example for an MPD:

- 23 • **MPD**@profiles="urn:mpeg:dash:profile:isoff-on-demand:2011,  
24 http://dashif.org/guidelines/dash264, http://dashif.org/guide-  
25 lines/dashif#extension-example"
  - 26 ○ @id = 1, **AdaptationSet**@profiles="urn:mpeg:dash:profile:isoff-on-  
27 demand:2011, http://dashif.org/guidelines/dash264"
  - 28 ○ @id = 2, **AdaptationSet**@profiles="http://dashif.org/guide-  
29 lines/dash264"
  - 30 ○ @id = 3, **AdaptationSet**@profiles="urn:mpeg:dash:profile:isoff-on-  
31 demand:2011, http://dashif.org/guidelines/dashif#extension-ex-  
32 ample"

33 Pruning process for profile <http://dashif.org/guidelines/dash264> results in

- 34 • **MPD**@profiles="http://dashif.org/guidelines/dash264"
  - 35 ○ @id = 1, **AdaptationSet**@profiles="http://dashif.org/guide-  
36 lines/dash264"
  - 37 ○ @id = 2, **AdaptationSet**@profiles="http://dashif.org/guide-  
38 lines/dash264"

39 It is now required that the pruned MPD conforms to DASH-IF IOP.

40 Pruning process for profile <http://dashif.org/guidelines/dashif#extension-example>  
41 results in

- 42 • **MPD**@profiles="http://dashif.org/guidelines/dash264"
  - 43 ○ @id = 3, **AdaptationSet**@profiles="http://dashif.org/guide-  
44 lines/dashif#extension-example"

45 It is now required that the pruned MPD conforms to DASH-IF Example Extension Adaptation  
46 Set.

---

# 1 **Annex B Live Services - Use Cases and Architecture**

## 2 **B.1 Baseline Use cases**

### 3 **B.1.1 Use Case 1: Live Content Offered as On-Demand**

4 In this case content that was distributed as live is offered in a separate Media Presentation as  
5 On-Demand Content.

### 6 **B.1.2 Use Case 2: Scheduled Service with known duration and Operating 7 at live edge**

8 In this case a service started a few minutes ago and lasts 30 minutes. The duration is known  
9 exactly and also all segment URLs are known. The timeshift buffer is short. This may for ex-  
10 ample be a live service for which the service provider wants to ensure that only a small window  
11 is accessible. The content is typically be pre-canned, but offered in a scheduled manner.

### 12 **B.1.3 Use Case 3: Scheduled Service with known duration and Operating 13 at live edge and time shift buffer**

14 In this case a service started a few minutes ago and lasts 30 minutes. The duration is known  
15 exactly and also all segment URLs are known. The timeshift buffer is long. This may for ex-  
16 ample be a service for which the service provider wants to ensure that the content is made  
17 available in a scheduled manner, e.g. no client can access the content earlier than scheduled by  
18 the content provider. However, after the live edge is completed, the content is available for  
19 24h. The content is typically pre-canned.

### 20 **B.1.4 Use Case 4: Scheduled Live Service known duration, but unknown 21 Segment URLs**

22 In this case a live service started a few minutes ago and lasts 30 minutes. The duration is known  
23 exactly but the segment URLs are unknown, as for example some advertisement may be added  
24 on the fly. Otherwise this service is similar to use case 3.

### 25 **B.1.5 Use Case 5: 24/7 Live Service**

26 In this case a live service started that may have started a long time ago is made available. Ad  
27 breaks and operational updates may be done with a 30sec pre-warning. The duration is un-  
28 known and also the segment URLs, the exact set of provided media components (different  
29 language tracks, subtitles, etc.) are unknown, as for example some advertisement may be added  
30 on the fly. Otherwise this service is similar to use case 3.

### 31 **B.1.6 Use Case 6: Approximate Media Presentation Duration Known**

32 In this case a live service starts at a specific time. The duration is known approximately and  
33 also all segment URLs are known for the approximate duration. Towards the end of the Media  
34 Presentation, the Media Presentation duration may be extended or may be finally determined  
35 by providing an update of the MPD.

36

## B.2 Baseline Architecture for DASH-based Live Service

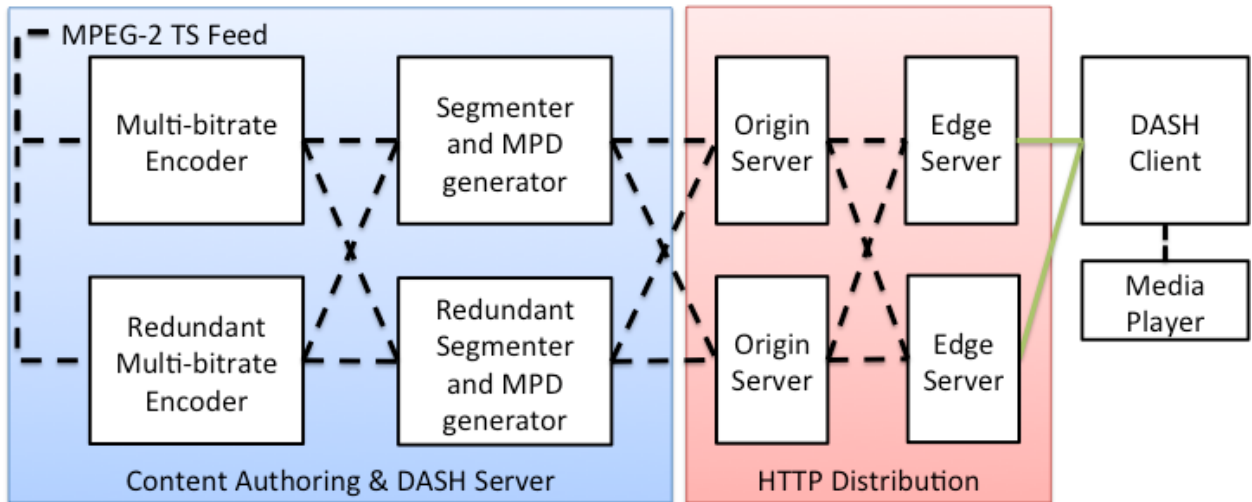


Figure 22 Typical Deployment Scenario for DASH-based live services

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.

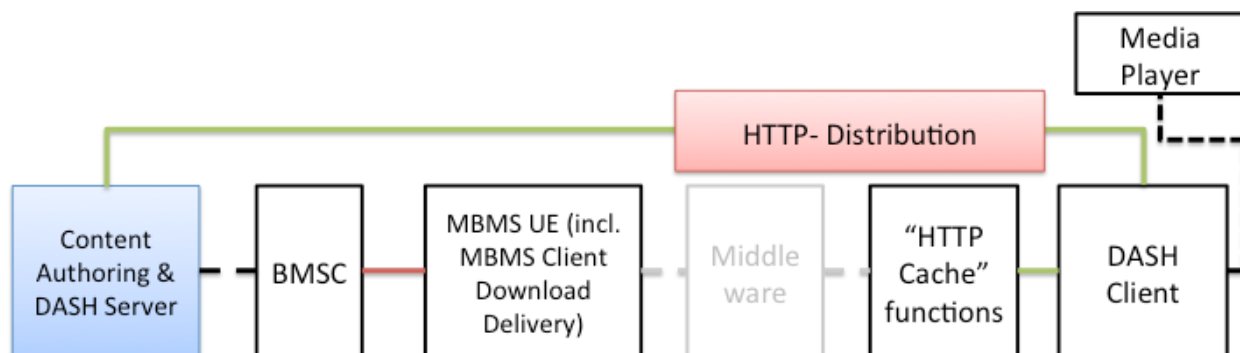
An MPEG2-TS stream is used often as feed into the encoder chain. The multi-bitrate encoder produces the required number of Representations for each media component and offers those in one Adaptation Set. In the context of this document is assumed that content is offered in the ISO BMFF live profile with 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 section 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.



1  
2  
3 **Figure 23 Typical Deployment Scenario for DASH-based live services partially offered through MBMS (unidirectional FLUTE distribution)**

4 The same content authoring and DASH server solution as shown in Figure 1 are considered in  
5 this baseline architecture. The DASH Segmenter (cf. Fig .1) provides DASH segments of typ-  
6 ically one quality representation into the BM-SC, which sends the segments using MBMS  
7 Download (as sequence of files using IETF FLUTE protocol) to the MBMS User Equipment  
8 (UE). The MBMS UE includes the needed MBMS download delivery client functions to re-  
9 cover the media segments from the FLUTE reception. The MBMS UE makes the segments  
10 through a local HTTP Cache function available to the DASH client. The DASH client uses  
11 HTTP (green line) to retrieve the segments from the device local cache.

12 In case the MBMS reception is not possible for that Video Session, the DASH client can use  
13 unicast HTTP to acquire the stream (according to previous section).

14 Note, the objective of the client architecture realization here is on using a generic DASH client  
15 for unicast and broadcast. More customized implementations are possible.

## 16 **B.4 Typical Problems in Live Distribution**

### 17 **B.4.1 Introduction**

18 Based on the deployment architectures in Figure 22 and Figure 23 a few typical problems in  
19 DASH-based ABR distribution are explained.

### 20 **B.4.2 Client Server Synchronization Issues**

21 In order to access the DASH segments at the proper time as announced by the segment availa-  
22 bility times in the MPD, client and server need to operate in the same time source, in general a  
23 globally accurate wall-clock, for example provided by NTP or GPS. There are different reasons  
24 why the DASH client and the media generation source may not have identical time source,  
25 such as

- 26 • DASH client is off because it does not have any protocol access to accurate timing.  
27 This may for example be the case for DASH clients that are running in the browser or  
28 on top of a general-purpose HTTP stack.
- 29 • DASH client clock drifts against the system clock and the DASH client is not synchro-  
30 nizing frequently enough against the time-source.
- 31 • The segmenter synchronized against a different time source than DASH client.
- 32 • There may be unknown delay on the ingest to the server/cache whether the segment is  
33 accessible. This is specifically relevant if MBMS is used as the contribution link result-  
34 ing in transport delay.
- 35 • It may also be that the MPD provides the availability times at the segmenter, but the  
36 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 22 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 trans-coded 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.

---

## 1 **B.4.6 Swapping across Redundant Tools**

2 In case of failures, redundant tools kick in. If the state is not fully maintained across redundant  
3 tools, the service may not be perceived continuous by DASH client. Problems that may happen  
4 at the encoder, that redundant encoders do not share the same timeline or the timeline is inter-  
5 rupted. Depending on the swap strategy ("hot" or "warm"), the interruptions are more or less  
6 obvious to the client. Similar issues may happen if segmenters fail, for example the state for  
7 segment numbering is lost.

## 8 **B.4.7 CDN Issues**

9 Typical CDN operational issues are the following:

- 10 • Cache Poisoning – at times segment generation may be erroneous. The encoder can  
11 produce a corrupt segment, or the segment can become corrupted during upload to  
12 origin. This can happen for example if encoder connectivity fails in mid segment up-  
13 load, leading to a malformed segment (with the correct name) being sent to edge and  
14 caching servers. The CDN then caches this corrupt segment and continues to deliver it  
15 to fulfill future requests, leading to widespread client failures.
- 16 • Cache inconsistency – with a dual origin scheme, identically named segments can be  
17 produced with slight differences in media time, due to clock drift or other encoder is-  
18 sues. These segments are then cached by CDNs and used to respond to client requests.  
19 If segments from one encoder are mixed with segments of another, it can lead to dis-  
20 continuous playback experiences on the clients.

## 21 **B.4.8 High End-to-end Latency**

22 End-to-end latency (also known as hand-waving latency) is defined as the accumulated delay  
23 between an action occurring in front of the camera and that action being visible in a buffered  
24 player. It is the sum of

- 25 1. Encoder delay in generating a segment.
- 26 2. Segment upload time to origin server from the encoder.
- 27 3. Edge server segment retrieval time from origin
- 28 4. Segment retrieval time by the player from the edge server
- 29 5. The distance back from the live point at which the player chooses to start playback.
- 30 6. Buffering time on the player before playback commences.

31 In steps 1 through 4, assuming non-chunked HTTP transfer, the delay is a linear function of  
32 the segment duration. Overly conservative player buffering can also introduce unnecessary de-  
33 lay, as can choosing a starting point behind the live point. Generally the further behind live the  
34 player chooses to play, the more stable the delivery system is, which leads to antagonistic de-  
35 mands on any production system of low latency and stability.

## 36 **B.4.9 Buffer Management & Bandwidth Estimation**

37 The main user experience degradations in video streaming are rebuffering events. At the same  
38 time, user experience is influenced by the quality of the video (typically determined by the  
39 bitrate) as well as at least for certain cases on the end-to-end latency. In order to request the  
40 access bitrate, the client does a bandwidth estimation typically based on the history and based  
41 on this and the buffer level in the client it decides to maintain or switch Representations.

42 In order to compensate bandwidth variations, the client buffers some media data prior to play-  
43 out. More time buffer results less buffer under runs and less rebuffering, but increases end-to-  
44 end latency. In order to maximize the buffer in the client and minimize the end-to-end latency

---

1 the DASH client would like to request the media segment as close as possible to its actual  
2 segment availability start time. However, this may cause issues in the playout as the in case of  
3 bitrate variations, the buffer may drain quickly and result in playout starvation and rebuffering.

#### 4 **B.4.10 Start-up Delay and Synchronization Audio/Video**

5 At the start-up and joining, it is relevant that the media playout is initiated, but that the delay  
6 at start is reasonable and that the presentation is enabled such that audio and video are presented  
7 synchronously. As audio and video Representations typically are offered in different sampling  
8 rates, and segments of audio and video are not aligned at segment boundaries. Hence, for proper  
9 presentation at startup, it is necessary that the DASH client schedules the presentation at the  
10 presentation time aligned to the over media presentation timeline.

### 11 **B.5 Advanced Use Cases**

#### 12 **B.5.1 Introduction**

13 Based on the above issues a few advanced use cases are considered.

#### 14 **B.5.2 Use Case 7: Live Service with undetermined end**

15 In this case a live service started that may have started a long time ago is made available. The  
16 MPD update may be done with a 30sec pre-warning. The duration is unknown exactly but the  
17 segment URLs are unknown, as for example some advertisement may be added on the fly.  
18 Otherwise this service is similar to use case 3.

#### 19 **B.5.3 Use Case 8: 24/7 Live Service with canned advertisement**

20 In this case a live service started that may have started a long time ago is made available. The  
21 MPD update may be done with a 30sec pre-warning. The duration is unknown exactly but the  
22 segment URLs are unknown, as for example some advertisement may be added on the fly. The  
23 advertisement itself is not a dynamic service, but available on a server as a pre-canned adver-  
24 tisement.

#### 25 **B.5.4 Use case 9: 24x7 live broadcast with media time discontinuities**

26 In other use cases, the content provider splices content such as programs and ads with inde-  
27 pendent media timelines at the content provider.

#### 28 **B.5.5 Use case 10: 24x7 live broadcast with Segment discontinuities**

29 Based on the discussions above, interruptions in encoding, etc., but presentation and media  
30 timelines resume after loss of some segments.

31