INTERNATIONAL STANDARD

ISO/IEC 23009-1

Second edition
2014-05-15

Information technology — Dynamic adaptive streaming over HTTP (DASH) —

Part 1: Media presentation description and segment formats

Technologies de l'information — Diffusion en flux adaptatif dynamique sur HTTP (DASH) —

Partie 1: Description de la présentation et formats de remise des médias

Reference number
ISO/IEC 23009-1:2014(E)

COPYRIGHT PROTECTED DOCUMENT

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 23009-1 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

This second edition cancels and replaces the first edition (ISO/IEC 23009-1:2012), which has been technically revised. It also incorporates the Technical Corrigendum ISO/IEC 23009-1:2012/Cor.1:2013.

ISO/IEC 23009 consists of the following parts, under the general title *Information technology — Dynamic adaptive streaming over HTTP (DASH)*:

— *Part 1: Media presentation description and segment formats*

— *Part 2: Conformance and reference software*

— *Part 3: Implementation guidelines* [Technical Report]

— *Part 4: Segment encryption and authentication*

Introduction

Dynamic Adaptive Streaming over HTTP (DASH) is intended to support a media-streaming model for delivery of media content in which control lies exclusively with the client. Clients may request data using the HTTP protocol from standard web servers that have no DASH-specific capabilities. Consequently, this part of ISO/IEC 23009 focuses not on client or server procedures but on the data formats used to provide a DASH Media Presentation.

This part of ISO/IEC 23009 primarily specifies formats for the Media Presentation Description and Segments. It is applicable to streaming services over the Internet.

Information technology — Dynamic adaptive streaming over HTTP (DASH) —

Part 1:
Media presentation description and segment formats

1 Scope

This part of ISO/IEC 23009 primarily specifies formats for the Media Presentation Description and Segments for dynamic adaptive streaming delivery of MPEG media over HTTP. It is applicable to streaming services over the Internet.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ITU-T Rec. H.222.0 | ISO/IEC 13818-1, *Information technology — Generic coding of moving pictures and associated audio information: Systems*

ISO/IEC 14496-10, *Information technology — Coding of audio-visual objects — Part 10: Advanced Video Coding*

ISO/IEC 14496-12, *Information technology — Coding of audio-visual objects — Part 12: ISO base media file format* (technically identical to ISO/IEC 15444-12)

ISO/IEC 23001-8, *Information technology — MPEG systems technologies — Part 8: Coding-independent code points*

IETF RFC 2141, *URN Syntax*, May 1997

IETF RFC 2616, *Hypertext Transfer Protocol – HTTP/1.1*, June 1999

IETF RFC 3023, *XML Media Types*, January 2001

IETF RFC 3406, *Uniform Resource Names (URN) Namespace Definition Mechanisms*, October 2002

IETF RFC 3629, *UTF-8, a transformation format of ISO 10646*, November 2003

IETF RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax*, January 2005

IETF RFC 4122, *A Universally Unique IDentifier (UUID) URN Namespace*, July 2005

IETF RFC 4288, *Media Type Specifications and Registration Procedures*, December 2005

IETF RFC 4337, *MIME Type Registration for MPEG-4*, March 2006

IETF RFC 4648, *The Base16, Base32, and Base64 Data Encodings*, October 2006

IETF RFC 5261, An Extensible Markup Language (XML) Patch Operations Framework Utilizing XML Path Language (XPath) Selectors, September 2008

IETF RFC 5646, *Tags for Identifying Languages*, September 2009

IETF RFC 6265, *HTTP State Management Mechanism*, April 2011

IETF RFC 6381, *The 'Codecs' and 'Profiles' Parameters for "Bucket" Media Types*, August 2011

W3C XLINK *XML Linking Language (XLink) Version 1.1*, W3C Recommendation 06, May 2010

W3C Media Fragments URI 1.0 (basic), W3C Recommendation, 25 September 2012

3 Terms, definitions, symbols and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1.1
access unit
unit of a media stream with an assigned Media Presentation time

3.1.2
accessibility
degree to which a media content or certain media content components are available to as many people as possible

3.1.3
Adaptation Set
set of interchangeable encoded versions of one or several media content components

3.1.4
asset
content including media and metadata together with the rights to use the content by the content provider

3.1.5
available Segment
Segment that is accessible at its assigned HTTP-URL and a possibly assigned byte range that is the request with an HTTP GET results in a reply of the Segment and 2xx status code

3.1.6
Bitstream Switching Segment
Segment that if present contains essential data to switch to the Representation it is assigned to

3.1.7
complementary Representation
Representation which complements at least one dependent Representation

3.1.8
continuous media
media with an inherent notion of time, for example, speech, audio, video, timed text or timed metadata

3.1.9
DASH metric
metric identified by a key and defined in this part of ISO/IEC 23009

3.1.10
dependent Representation
Representation for which Segments from its complementary Representations are necessary for presentation and/or decoding of the contained media content components

3.1.11
earliest presentation time
smallest presentation time of any access unit of a Media Segment or Subsegment for a media stream

3.1.12
event
aperiodic sparse media-time related auxiliary information to the DASH client or to an application

3.1.13
event stream
sequence of related events

3.1.14
group
collection of Adaptation Sets that are not expected to be presented simultaneously

3.1.15
HTTP-URL
URL with a fixed scheme of “http” or “https”

3.1.16
Index Segment
Segment that primarily contains indexing information for Media Segments

3.1.17
Initialization Segment
Segment containing metadata that is necessary to present the media streams encapsulated in Media Segments

3.1.18
media content
one media content period or a contiguous sequence of media content periods

3.1.19
media content component
one continuous component of the media content with an assigned media component type that can be encoded individually into a media stream

3.1.20
media content component type
single type of media content such as audio, video, or text

3.1.21
media content period
set of media content components that have a common timeline as well as relationships on how they can be presented

3.1.22
Media Presentation
collection of data that establishes a bounded or unbounded presentation of media content

3.1.23
Media Presentation Description
MPD
formalized description for a Media Presentation for the purpose of providing a streaming service

3.1.24
Media Presentation timeline
concatenation of the timeline of all Periods which itself is common to all Representations in the Period

3.1.25
Media Segment
Segment that complies with media format in use and enables playback when combined with zero or more preceding segments, and an Initialization Segment (if any)

3.1.26
media stream
encoded version of a media content component

3.1.27
Media Subsegment
Subsegment that only contains media data but no Segment Index

3.1.28
message
part of an event containing information that is exclusively handled by the event handler

3.1.29
MPD start time
approximate presentation start time of a Media Segment signalled in MPD

3.1.30
MPD duration
approximate presentation duration of a Media Segment signalled in MPD

3.1.31
Period
interval of the Media Presentation, where a contiguous sequence of all Periods constitutes the Media Presentation

3.1.32
presentation time
time associated to an access unit that maps it to the Media Presentation timeline

3.1.33
remote element entity
entity that contains one or more elements and is referenced in the MPD with an HTTP-URL contained in an `@xlink:href` attribute

3.1.34
Representation
collection and encapsulation of one or more media streams in a delivery format and associated with descriptive metadata

3.1.35
Segment
unit of data associated with an HTTP-URL and optionally a byte range that are specified by an MPD

3.1.36
Segment availability start time
latest time instant in wall-clock time at which a Segment becomes an available Segment

3.1.37
adjusted Segment availability start time
time instant in wall-clock time at which a Segment becomes an available Segment

3.1.38
Segment availability end time
time instant in wall-clock time at which a Segment ceases to be an available Segment

3.1.39
Segment Index
compact index of the time range to byte range mapping within a Media Segment separately from the MPD

3.1.40
stream access point
SAP
position in a Representation enabling playback of a media stream to be started using only the information contained in Representation data starting from that position onwards (preceded by initializing data in the Initialization Segment, if any)

3.1.41
Sub-Representation
part of a Representation described in the MPD that is present in the entire Period

3.1.42
Subsegment
unit within Media Segments that is indexed by a Segment Index

3.1.43
valid Segment URL
HTTP-URL that is promised to reference a Segment during its Segment availability period

3.1.44
wall-clock time
time as stated by UTC

3.2 Symbols and abbreviated terms

For the purposes of this document, the following symbols and abbreviated terms apply.

AVC advanced video coding

CAT conditional access table

DASH dynamic adaptive streaming over HTTP

DM DASH Metrics

DRM digital rights management

ECM entitlement control message

HTTP hypertext transfer protocol

IDR instantaneous decoding refresh

ISOBMFF	ISO base media file format
MPD	Media Presentation Description
MVC	multi-view video coding
PAT	program association table
PCR	program clock reference
PES	packetized elementary stream
PID	packet identifier
PMT	program map table
PSI	program specific information
PTS	presentation time stamp
SAP	stream access point
SEI	supplementary enhancement information
SVC	scalable video coding
TCP	transmission control protocol
TLS	transport layer security
TS	transport stream
URI	uniform resource identifier
URL	uniform resource locator
URN	uniform resource name
UTC	coordinated universal time
UUID	universally unique identifier
XML	extensible mark-up language

3.3 Conventions

The following naming conventions apply in this document.

— Elements in an XML document are identified by an upper-case first letter and in bold face as **`Element`**. To express that an element **`Element1`** is contained in another element **`Element2`**, we may write **`Element2.Element1`**. If an element's name consists of two or more combined words, camel-casing is typically used, e.g. **`ImportantElement`**. Elements may be present either exactly once, or the minimum and maximum occurrence is defined by <minOccurs> ... <maxOccurs>.

— Attributes in an XML document are identified by a lower-case first letter as well as they are preceded by a '`@`'-sign, e.g. `@attribute`. To point to a specific attribute `@attribute` contained in an element **`Element`**, one may write **`Element`**`@attribute`. If an attribute's name consists of two or more combined words, camel-casing is typically used after the first word, e.g.

`@veryImportantAttribute`. Attributes may have assigned a status in the XML as mandatory (M), optional (O), optional with default value (OD) and conditionally mandatory (CM).

— Namespace qualification of elements and attributes is used as per XML standards, in the form of **`namespace:Element`** or `@namespace:attribute` The fully qualified namespace will be provided in the schema fragment associated with the declaration. External specifications extending the namespace of DASH are expected to document the element name in the semantic table with an extension namespace prefix.

— Variables defined in the context of this document are specifically highlighted with *italics,* e.g. *InternalVariable*.

— Structures that are defined as part of the hierarchical data model are identified by an upper-case first letter, e.g. Period, Adaptation Set, Representation, Segment, etc.

— The term "this clause" refers to the entire clause included within the same first heading number. The term "this subclause" refers to all text contained in the subclause with the lowest hierarchy heading.

4 Introduction

4.1 System description

Dynamic Adaptive Streaming over HTTP (DASH) specifies XML and binary formats that enable delivery of media content from standard HTTP servers to HTTP clients and enable caching of content by standard HTTP caches.

This part of ISO/IEC 23009 primarily defines two formats:

— The Media Presentation Description (MPD) describes a *Media Presentation*, i.e. a bounded or unbounded presentation of media content. In particular, it defines formats to announce resource identifiers for *Segments* and to provide the context for these identified resources within a Media Presentation. These resource identifiers are HTTP-URLs possibly combined with a byte range.

— The Segment formats specify the formats of the entity body of the HTTP response to an HTTP GET request or a partial HTTP GET with the indicated byte range using HTTP/1.1 as defined in RFC 2616 to a resource identified in the MPD. Segments typically contain efficiently coded media data and metadata conforming to or at least closely aligned with common media formats.

The MPD provides sufficient information for a client to provide a streaming service to the user by accessing the Segments through the protocol specified in the scheme of the defined resources. In the context of this part of ISO/IEC 23009 the assumed protocol is HTTP/1.1. Such a client is referred to as a DASH Client in the remainder of 23009-1. However, this part of ISO/IEC 23009 does not provide a normative specification for such a client.

Figure 1 shows a possible deployment architecture in which the formats defined in this part of ISO/IEC 23009 may be used. Boxes with solid lines indicate devices that are mentioned in this specification as they host or process the formats defined in this specification whereas dashed boxes are conceptual or transparent. This part of ISO/IEC 23009 deals with the definition of formats that are accessible on the interface to the DASH Client, indicated by the solid lines. Any other formats or interfaces are not in scope of this Part of ISO/IEC 23009. In the considered deployment scenario, it is assumed that the DASH Client has access to an MPD. The MPD provides sufficient information for the DASH Client to provide a streaming service to the user by requesting Segments from an HTTP server and demultiplexing, decoding and rendering the included media streams.

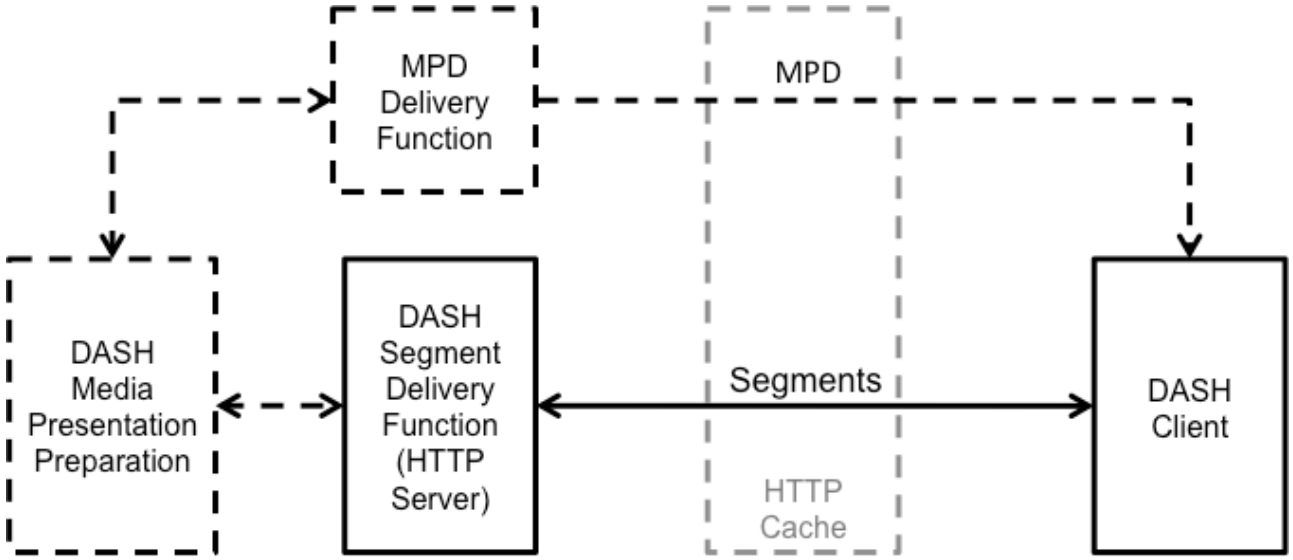

Figure 1 — Example system for DASH formats

Note that despite the formats are initially designed to be used in the above deployment scenario their application is obviously not restricted to this scenario. The particular aspect on "HTTP" in DASH is the usage of HTTP-URLs in the MPD for the purpose to refer to Segments. The usage of HTTP-URLs enables unique location information and it provides well-defined methods to access the resources, in particular HTTP GET and HTTP partial GET.

4.2 DASH client model

The design of the formats defined in this part of ISO/IEC 23009 is based on the informative client model as shown in Figure 2. The figure illustrates the logical components of a conceptual DASH client model. In this figure the DASH access engine receives the Media Presentation Description (MPD), constructs and issues requests and receives Segments or parts of Segments. In the context of this part of ISO/IEC 23009, the output of the DASH access engine consists of media in MPEG container formats (ISO/IEC 14496-12 ISO Base Media File Format or ISO/IEC 13818-1 MPEG-2 Transport Stream), or parts thereof, together with timing information that maps the internal timing of the media to the timeline of the Media Presentation. In Annex F of this part of ISO/IEC 23009, guidance on enabling the use of this part of ISO/IEC 23009 with other container formats is provided. In addition, the DASH access client may also receive and extract Events that are related to the media time. The events may be processed in the DASH client or may be forwarded to an application in the execution environment of the DASH client.

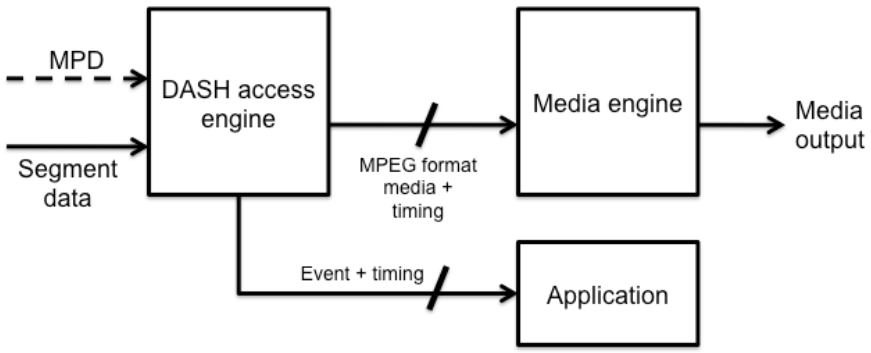

Figure 2 — DASH Client Model

4.3 DASH data model overview

DASH is intended to support a media-streaming model for delivery of media content in which control lies primarily with the client. Clients may request data using the HTTP protocol from standard web servers that have no DASH-specific capabilities. Consequently, this standard focuses not on client or server procedures but on the data formats used to provide a DASH Media Presentation.

The collection of encoded and deliverable versions of media content and the appropriate description of these form a Media Presentation. Media content is composed of a single or multiple contiguous media content **periods** in time. Content in different media content periods may be completely independent or certain periods of a Media Presentation may belong to the same Asset, for example a Media Presentation is a collection of main program composed of multiple periods, each assigned to the same Asset, and interleaved with inserted advertisement periods. Each media content period is composed of one or multiple **media content components,** for example audio components in various languages, different video components providing different views of the same program, subtitles in different language, etc.. Each media content component has an assigned **media content component type**, for example audio or video.

Each media content component may have several encoded versions, referred to as **media streams**. Each media stream inherits the properties of the media content, the media content period, the media content component from which it was encoded and in addition it gets assigned the properties of the encoding process such as sub-sampling, codec parameters, encoding bitrate, etc. This describing metadata is relevant for static and dynamic selection of media content components and media streams.

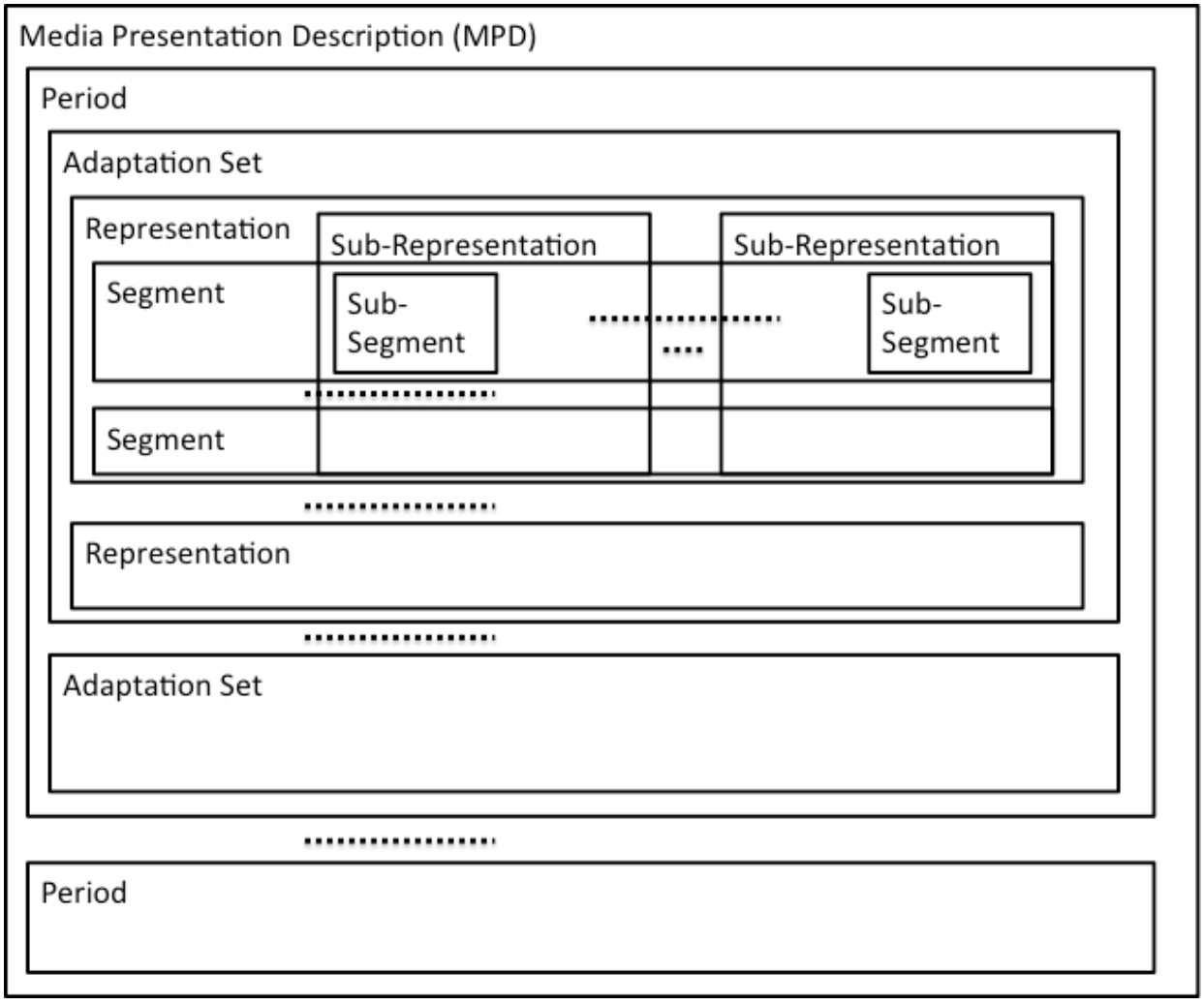

Figure 3 — DASH High-Level Data Model

DASH is based on a hierarchical data model aligned with the presentation in Figure 3. A DASH Media Presentation is described by a **Media Presentation Description** document. This describes the sequence of **Periods** (see 5.3.2) in time that make up the Media Presentation. A Period typically represents a media content period during which a consistent set of encoded versions of the media content is available i.e. the set of available bitrates, languages, captions, subtitles etc. does not change during a Period.

Within a Period, material is arranged into **Adaptation Sets** (see 5.3.3). An Adaptation Set represents a set of interchangeable encoded versions of one or several media content components (see 5.3.4). For example there may be one Adaptation Set for the main video component and a separate one for the main audio component. If there is other material available, for example captions or audio descriptions, then these may each have a separate Adaptation Set. Material may also be provided in multiplexed form, in which case interchangeable versions of the *multiplex* may be described as a single Adaptation Set, for example an Adaptation Set containing both the main audio and main video for a Period. Each of the multiplexed components may be described individually by a media content component description.

An Adaptation Set contains a set of **Representations** (see 5.3.5). A Representation describes a *deliverable encoded version* of one or several media content components. A Representation includes one or more media streams (one for each media content component in the multiplex). Any single Representation within an Adaptation Set is sufficient to render the contained media content components. By collecting different Representations in *one* Adaptation Set, the Media Presentation author expresses that the Representations represent perceptually equivalent content. Typically this means, that clients may switch dynamically from Representation to Representation within an Adaptation Set in order to adapt to network conditions or other factors. Switching refers to the presentation of decoded data up to a certain time t, and presentation of decoded data of another Representation from time t onwards. If Representations are included in one Adaptation Set, and the client switches properly, the Media Presentation is expected to be perceived seamless across the switch. Clients may ignore Representations that rely on codecs or other rendering technologies they do not support or that are otherwise unsuitable.

Within a Representation, the content may be divided in time into **Segments** (see 5.3.9 and 6) for proper accessibility and delivery. In order to access a Segment, a URL is provided for each Segment. Consequently, a Segment is the largest unit of data that can be retrieved with a single HTTP request.

NOTE This is not strictly true, since the MPD may also include a byte range with the URL, meaning that the Segment is contained in the provided byte range of some larger resource. An intelligent client could in principle construct a single request for multiple Segments, but this would not be the typical case.

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

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

Segments are assigned a duration, which is the duration of the media contained in the Segment when presented at normal speed. Typically all Segments in a Representation have the same or roughly similar duration. However Segment duration may differ from Representation to Representation. A DASH presentation can be constructed with relative short segments (for example a few seconds), or longer Segments including a single Segment for the whole Representation.

Short Segments are usually required in the case of live content, where there are restrictions on end-to-end latency. The duration of a Segment is typically a lower bound on the end-to-end latency. DASH does not

support the possibility for Segments to be extended over time: a Segment is treated as an object as a complete and discrete unit that must be made available in its entirety. However, this does not prevent to apply advanced HTTP transfer modes such as chunked transfer to optimize deployments and reduce end-to-end latency.

Segments may be further subdivided into **Subsegments** each of which contains a whole number of complete access units. There may also be media-format-specific restrictions on Subsegment boundaries, for example in the ISO Base Media File Format a Subsegment must contain a whole number of complete movie fragments. If a Segment is divided into Subsegments they are described by a compact **Segment index**, which provides the presentation time range in the Representation and corresponding byte range in the Segment occupied by each Subsegment. Clients may download this index in advance and then issue requests for individual Subsegments.

Clients may switch from Representation to Representation within an Adaptation Set at any point in the media. However, switching at arbitrary positions may be complex because of coding dependencies within Representations and other factors, potentially requiring parallel download and decoding in the DASH client. It is also desirable to avoid download of 'overlapping' data i.e. media for the same time period from multiple Representations. Usually, switching is simplest at a Stream Access Point (SAP) in the new stream. In order to formalize requirements related to switching DASH defines a codec-independent concept of Stream Access Points and identifies various types of Stream Access Points.

Segmentation and Subsegmentation may be performed in ways that make switching simpler. For example, in the very simplest cases each Segment or Subsegment begins with a SAP and the boundaries of Segments or Subsegments are aligned across the Representations of one Adaptation Set. In this case, switching Representation involves playing to the end of a (Sub)Segment of one Representation and then playing from the beginning of the next (Sub)Segment of the new Representation. The Media Presentation Description and Segment Index provide various indications, which describe properties of the Representations that may make switching simpler. Profiles of this specification may then require these indicators to be set in certain ways, making implementation of clients for those profiles simpler at the cost of requiring the media data to obey the indicated constraints.

For On-Demand services, the Media Presentation Description is typically a static document describing the various aspects of the Media Presentation. All Segments of the Media Presentation are available on the server once any Segment is available. For live services, however, Segments become available with time as the content is produced and therefore, dynamic Media Presentations are suitable. The Media Presentation Description may be updated regularly to reflect changes in the presentation over time, for example Segment URLs for new segments may be added to the MPD and those for old, no longer available Segments may be removed. However, if Segment URLs are described using a template, this updating may not be necessary except for some redundancy/failover cases.

Events may be provided in the MPD or within a Representation in order to signal aperiodic information to the DASH client or to an application. Events are timed, i.e. each event starts at a specific media presentation time and typically has duration. Events include DASH specific signalling or application-specific events. Examples for events are indication of MPD updates on the server, possibly providing the detailed update as part of the messages. The event mechanisms may also be used to deliver media time related application events, for example information about ad insertion opportunities, etc.

4.4 Protocols

This Part of ISO/IEC 23009 may be deployed in a system according to Figure 1 for which

— The DASH Client includes a *client* as specified in RFC 2616 and,

— The HTTP Server hosting the DASH Segments complies with a *server* as specified in RFC 2616.

DASH Clients typically use the HTTP GET method or the HTTP partial GET method, as specified in RFC 2616, Clause 9.3, to access Segments or parts thereof.

The use of HTTP as a transport protocol inherently provides many advanced features such as caching, redirection or authentication. As another example, transport security in HTTP-based delivery may be achieved by using HTTP over TLS as specified in RFC 2818. Yet another example is the use of HTTP state management mechanisms (also known as Cookies) as defined in RFC 6265.

However, the formats defined in this part of ISO/IEC 23009 may also be used with other protocols. In particular, the objects may be delivered with any object delivery protocol that provides a binding between an HTTP-URL and the delivered object.

4.5 Media Stream and Representation properties

4.5.1 Switching and Random Access Support

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

For this purpose this subsection defines three relevant concepts to support seamless switching, namely

- Media stream access points in 4.5.2 to signal positions where to easily switch to a Representation, and in addition where to suitable access a Representation at start-up or seek.
- Non-overlapping Segments and Subsegments in 4.5.3 to signal that at the signalled stream access points, no overlap decoding of Representations is necessary in order to provide a continuous switch
- Bitstream concatenation in 4.5.4 to signal that the concatenation of two Representations at a switch point results in a conforming bitstream

These three properties are neither sufficient nor necessary for seamless switching, but certain implementation or profiles may use these properties in order to simplify practical implementations.

4.5.2 Media stream access points

To be able to access a Representation, each of the media streams that are contained in the Representation requires Media Stream Access Points (SAPs). SAPs in the context of this part of ISO/IEC 23009 refer to the SAP definition in ISO/IEC 14496-12 Annex I. ISO/IEC 14496-12 Annex I.3 defines different types of SAPs that provide a relationship between the position where a stream can be accessed, relative to the start of a Segment or Subsegment, its presentation time and the presentation times and position of other access unit in the stream. The same SAP type definitions shall apply for this part of ISO/IEC 23009.

A SAP is a position in a Representation that enables playback of a media stream to be started using only the information contained in Representation data starting from that position onwards (preceded by initializing data in the Initialization Segment, if any).

For each SAP the properties, I_{SAP}, T_{SAP}, I_{SAU}, T_{DEC}, T_{EPT}, and T_{PTF} are identified and defined in ISO/IEC 14496-12 Annex I.2.

In particular, T_{SAP} is defined to be earliest presentation time of any access unit of the media stream such that all access units of the media stream with presentation time greater than or equal to T_{SAP} can be correctly decoded using data in the Representation starting at byte position I_{SAP} and no data before I_{SAP}.

NOTE The type of SAP is dependent only on which Access Units are correctly decodable and their arrangement in presentation order. The types informally correspond with some common terms:

- Type 1 corresponds to what is known in some coding schemes as a “Closed GoP random access point” (in which all access units, in decoding order, starting from I_{SAP} can be correctly decoded, resulting in a continuous time sequence of correctly decoded access units with no gaps) and in addition the access unit in decoding order is also the first access unit in presentation order.

- Type 2 corresponds to what is known in some coding schemes as a “Closed GoP random access point”, for which the first access unit in decoding order in the media stream starting from I_{SAU} is not the first access unit in presentation order.

- Type 3 corresponds to what is known in some coding schemes as an “Open GoP random access point”, in which there are some access units in decoding order following I_{SAU} that cannot be correctly decoded and have presentation times less than T_{SAP}.

- Type 4 corresponds to what is known in some coding schemes as an "Gradual Decoding Refresh (GDR) random access point”, in which there are some access units in decoding order starting from and following I_{SAU} that cannot be correctly decoded and have presentation times less than T_{SAP}.

- Type 5 corresponds to the case for which there is at least one access unit in decoding order starting from I_{SAP} that cannot be correctly decoded and has presentation time greater than T_{DEC} and where T_{DEC} is the earliest presentation time of any access unit starting from I_{SAU}.

- Type 6 corresponds to the case for which there is at least one access unit in decoding order starting from I_{SAP} that cannot be correctly decoded and has presentation time greater than T_{DEC} and where T_{DEC} is not the earliest presentation time of any access unit starting from I_{SAU}.

SAPs are mostly relevant for two purposes in this part of ISO/IEC 23009:

1. For randomly accessing a Media Presentation, for example at the startup of the Media Presentation, after a seeking operation or after an error event especially in live cases.

2. To permit switching between two Representations whereby for seamless switching each media stream *i* in the switch-from Representation is presented up to $T_{SAP}(i)$ and each media stream *i* in the switch-to Representation is presented from the media Stream Access Point starting from $T_{SAP}(i)$.

There are obvious benefits for the client to be able to identify SAPs and one or several of their properties, in particular I_{SAP} and T_{SAP} for each media stream without requiring to access data at positions following I_{SAP}. DASH provides functionalities to explicitly signal such information by using signals in the MPD or the Segment Index or combinations of the two.

4.5.3 Non-overlapping Segments and Subsegments

Segments and Subsegments represent units for which the client has an exact map on how to access and download the unit using HTTP GET or HTTP partial GET methods.

Segments (respectively Subsegments) are typically generated by segmenting encoded media streams into appropriate units. If the generation of Segments (respectively Subsegments) adheres to certain rules, then the sequential decoding and presentation of Media Segments (respectively Subsegments) results in a correct presentation of all contained media streams. To define such rules the notion of “non-overlapping” segments (respectively Subsegments) is defined as follows.

Let

— $T_E(S,i)$ be the earliest presentation time of any access unit in stream *i* of a Segment or Subsegment *S*,

— $T_L(S,i)$ be the latest presentation time of any access unit in stream *i* of a Segment or Subsegment *S*.

Then two segments (respectively Subsegments), *A* and *B*, which may or may not be of different Representations, are *non-overlapping* if $T_L(A,i) < T_E(B,i)$ for all media streams *i* in A and B or if $T_L(B,i) < T_E(A,i)$ for all streams *i* in A and B where *i* refers to the same media component.

The property of “non-overlapping” segments (respectively Subsegments) is used to define the terms Segment alignment and Subsegment alignment.

4.5.4 Bitstream concatenation

A sequence of Segments (respectively Subsegments) is a “conforming Segment (respectively Subsegment) sequence” if the concatenation of all Segments (respectively Subsegments) in the sequence of Segments (respectively Subsegments) results in a bitstream that conforms to the media formats in use (including container and codecs).

NOTE This implies that a player conforming to the media format can play the resulting bitstream.

4.6 Brands

The ISO Base Media File Format, ISO/IEC 14496-12, defines the concept of brands; brand values identify specifications or conformance points. This part of ISO/IEC 23009 specifies several brands, as listed in Table 1.

Table 1 — Brands defined in this part of ISO/IEC 23009

Brand Identifier	Clause in this part of ISO/IEC 23009	Informative description
`emsg`	5.10.3.3	Event message box
`msdh`	6.3.4.2	Media Segment conforming to the general format type for ISO base media file format.
`msix`	6.3.4.3	Media Segment conforming to the Indexed Media Segment format type for ISO base media file format.
`sims`	6.3.4.4	Media Segment conforming to the Sub-Indexed Media Segment format type for ISO base media file format.
`dsms`	6.3.5.1	Media Segment conforming to the DASH Self-Initializing Media Segment format type for ISO base media file format.
`dash`	6.3.5.2	ISO base media file format file specifically designed for DASH including movie fragments and Segment Index.
`sisx`	6.4.6.2	Single Index Segment used to index MPEG-2 TS based Media Segments.
`risx`	6.4.6.3	Representation Index Segment used to index MPEG-2 TS based Media Segments.
`ssss`	6.4.6.4	Subsegment Index Segment used to index MPEG-2 TS based Media Segments.
`lmsg`	7.3.1	last Media Segment indicator for ISO base media file format.

4.7 Schemes

This part of ISO/IEC 23009 specifies several schemes as listed in Table 2.

Table 2 — Schemes defined in this part of ISO/IEC 23009

Scheme Identifier	Clause in this part of ISO/IEC 23009	Informative description
urn:mpeg:dash:schema:mpd:2011	Annex B	The namespace of the XML schema for the MPD.
urn:mpeg:dash:resolve-to-zero:2013	5.5.3	xlink resolution to zero element
urn:mpeg:dash:mp4protection:2011	5.8.5.2	protection schemes identified by a the Scheme Type within the Scheme Type Box of the Protection Scheme Information Box of ISO/IEC14496-12.
urn:mpeg:dash:13818:1:CA_descriptor:2011	5.8.5.2	Conditional Access System used for ISO/IEC 13818-1 (MPEG-2 Transport Stream).
urn:mpeg:dash:14496:10:frame_packing_arrangement_type:2011	5.8.5.3	frame-packing arrangement.
urn:mpeg:dash:13818:1:stereo_video_format_type:2011	5.8.5.3	frame-packing arrangement.
urn:mpeg:dash:23003:3:audio_channel_configuration:2011	5.8.5.4	channel configuration.
urn:mpeg:dash:outputChannelPositionList:2012	5.8.5.4	a list of output channel position to signal individual speaker positions as defined in ISO/IEC 23001-8.
urn:mpeg:dash:role:2011	5.8.5.5	DASH role scheme.
urn:mpeg:dash:stereoid:2011	5.8.5.6	scheme for multiple views media content description.
urn:mpeg:dash:event:2012	5.10.4	DASH event signalling scheme
urn:mpeg:dash:profile:full:2011	8.2	identifier for Full profile.
urn:mpeg:dash:profile:isoff-on-demand:2011	8.3	identifier for ISO Base media file format On Demand profile.
urn:mpeg:dash:profile:isoff-live:2011	8.4	identifier for ISO Base media file format live profile.
urn:mpeg:dash:profile:isoff-main:2011	8.5	identifier for ISO Base media file format main profile.
urn:mpeg:dash:profile:mp2t-main:2011	8.6	identifier for MPEG-2 TS main profile.
urn:mpeg:dash:profile:mp2t-simple:2011	8.7	identifier for MPEG-2 TS simple profile.

5 Media Presentation

5.1 General

A Media Presentation is a collection of data that is accessible to a DASH Client to provide a streaming service to the user.

A Media Presentation is described by an MPD including possible updates of the MPD. The MPD is defined in 5.2 and the MPD update mechanism is defined in 5.4. Assembly of a fragmented MPD is defined in 5.5. The data model that constitutes a Media Presentation is defined in 5.3. In 5.6, the formats and processing of URLs in the MPD is introduced. Program information is defined in 5.7. Descriptors associated to Representations or collections thereof are provided in 5.8. DASH metric collection description is specified in 5.9.

5.2 Media Presentation Description

5.2.1 General

The Media Presentation Description (MPD) is a document that contains metadata required by a DASH Client to construct appropriate HTTP-URLs to access Segments and to provide the streaming service to the user.

NOTE actual playback of the media streams included in the Representations is not controlled by the MPD information. Playback is controlled by the media engine operating on the media streams contained in the Representations in the usual way.

The format of URLs in the MPD and the process to generate HTTP GET and partial GET requests from URLs provided in the MPD is defined in 5.6.

The MPD is an XML document that is formatted according to the XML schema provided in Annex B. Some context on the schema is provided in 5.2.2.

The extension of the DASH XML schema (as provided in Annex B), in particular the addition of XML attributes or elements in the DASH namespace is reserved to ISO/IEC. Elements and attributes that have been added to the namespace compared to earlier revisions of this standard are documented in 5.2.3.

The MPD shall be authored such that, after XML attributes or elements in the DASH namespace but not in the XML schema documented in Annex B are removed, the result is a valid XML document formatted according to that schema and that conforms to this part of ISO/IEC 23009.

In addition, the MPD shall be authored such that, after XML attributes or elements in the other namespaces than the DASH namespace are removed, the result is a valid XML document formatted according to that schema and that conforms to this part of ISO/IEC 23009.

NOTE 1: Based on the last two paragraphs if DASH clients remove all XML attributes and elements from the MPD in the DASH namespace and in other namespaces that are not in the XML schema documented in Annex B, the MPD results in a valid XML document which complies with this part of ISO/IEC 23009. The DASH client may use such a resulting MPD for presentation of a conforming Media Presentation.

In addition, rules for authoring of MPDs conforming to a specific profile are provided in 8. Certain profiles as defined in 8 of this document may permit ignoring certain elements and attributes. However, this has no effect on the general MPD conformance rules defined in this subsection.

Following XML rules, the MPD document shall contain exactly one **`MPD`** element as specified in 5.3.

The MIME type of the MPD document is defined in Annex C.

The encoding of the MPD shall be UTF-8 as defined in IETF RFC 3629. All data provided in extension namespaces shall be UTF-8 as defined in IETF RFC 3629. If binary data needs to be added, it shall be included in Base64 as described in IETF RFC 4648 within a UTF-8 encoded element with a proper name space or identifier, such that an XML parser knows how to process or ignore it.

The delivery of the MPD is not in scope of this part of ISO/IEC 23009. However, if the MPD is delivered over HTTP, then the MPD document may be transfer encoded for transport, as described in RFC 2616.

NOTE 1 As an example the GZip algorithm as defined in RFC 1952 [3] may be used for Transfer-encoding.

NOTE 2 MPD encryption is not a normative part of this standard. However, if operating in an insecure environment and required by the content/service provider, elements and attributes of MPD may be encrypted to protect their confidentiality by using the syntax and processing rules specified in the "XML Encryption Syntax and Processing" by W3C [8].

NOTE 3 MPD integrity protection is not a normative part of this standard. However, if operating in an insecure environment and required by the content/service provider, the digital signing and verification procedures specified in the "XML Signature Syntax and Processing" by W3C [9] may be used to protect data origin authenticity and integrity of the MPD.

5.2.2 Schema

The initial part of the XML schema of the MPD is provided below, including namespace and other definitions. Specific types, elements and attributes are introduced in the remainder of this subclause. The complete normative MPD schema is provided in Annex B of this part of ISO/IEC 23009. In case of any inconsistencies the schema in Annex B takes precedence over the XML syntax snippets provided in this clause.

```
<?xml version="1.0"?>
<xs:schema targetNamespace="urn:mpeg:dash:schema:mpd:2011"
  attributeFormDefault="unqualified"
  elementFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns="urn:mpeg:dash:schema:mpd:2011">

  <xs:import namespace="http://www.w3.org/1999/xlink" schemaLocation="xlink.xsd"/>

  <xs:annotation>
    <xs:appinfo>Media Presentation Description</xs:appinfo>
    <xs:documentation xml:lang="en">
      This Schema defines the Media Presentation Description for MPEG-DASH.
    </xs:documentation>
  </xs:annotation>

  <!-- MPD: main element -->
  <xs:element name="MPD" type="MPDtype"/>

  ...

</xs:schema>
```

5.2.3 Elements and Attributes added in Revisions and Amendments

5.2.3.1 Overview

In amendments and revisions of the standard, the schema defined in Annex B may have been extended. In order to track this, section 5.2.3 tracks the addition of elements and attributes.

By following the rules in 5.2.1, a single MPD can be authored for clients that implement different versions of this standard.

5.2.3.2 Elements and Attributes added in this Revision

This revision adds the following elements and attributes to the schema defined in Annex B compared to the 2012 revision (ISO/IEC 23009-1:2012/Cor 1:2013) of this part of the standard:

— **MPD**@publishTime

- **Period.AssetIdentifier**
- **Period.EventStream**
- **RepresentationBase.InbandEventStream**
- **SegmentBase**@availabilityTimeOffset
- **SegmentBase**@availabilityTimeComplete
- **BaseURL**@availabilityTimeOffset
- **BaseURL**@availabilityTimeComplete
- **Subset**@id

5.3 Hierarchical data model

5.3.1 Introduction

5.3.1.1 Overview

A Media Presentation as described in the **MPD** consists of

- A sequence of one or more Periods as described in 5.3.2.
- Each Period contains one or more Adaptation Sets as described in 5.3.3. In case an Adaptation Set contains multiple media content components, then each media content component is described individually as defined in 5.3.4.
- Each Adaptation Set contains one or more Representations as described in 5.3.5.
- Adaptation Sets, Representations and Sub-Representations share common attributes and elements that are described in 5.3.7.
- Each Period may contain one or more Subsets that restrict combination of Adaptation Sets for presentation. Subsets are described in 5.3.8.
- Each Representation consists of one or more Segments described in 6. Segment Information is introduced in 5.3.9. Segments contain media data and/or metadata to access, decode and present the included media content. Representations may also include Sub-Representations as defined in 5.3.6 to describe and extract partial information from a Representation.
- Each Segment consists of one or more Subsegments. Subsegments are described in 6.2.3.2.

The summary of the semantics of the attributes and elements within an **MPD** element are provided in Table 3 of 5.3.1.2. The XML syntax of the **MPD** element is provided in 5.3.1.3.

5.3.1.2 Semantics

Table 3 — Semantics of `MPD` element

Element or Attribute Name	Use	Description
`MPD`		The root element that carries the Media Presentation Description for a Media Presentation.
`@id`	O	specifies an identifier for the Media Presentation. It is recommended to use an identifier that is unique within the scope in which the Media Presentation is published. If not specified, no MPD-internal identifier is provided. However, for example the URL to the MPD may be used as an identifier for the Media Presentation.
`@profiles`	M	specifies a list of Media Presentation profiles as described in 8. The contents of this attribute shall conform to either the `pro-simple` or `pro-fancy` productions of RFC6381, Section 4.5, without the enclosing `DQUOTE` characters, i.e. including only the `unencodedv` or `encodedv` elements respectively. As profile identifier the URI defined for the conforming Media Presentation profiles as described in 8 shall be used.
`@type`	OD default: `static`	specifies the type of the Media Presentation. For static Media Presentations (`@type="static"`) all Segments are available between the `@availabilityStartTime` and the `@availabilityEndTime`. For dynamic Media Presentations (`@type="dynamic"`) Segments typically have different availability times. For details refer to 5.3.9.5.3. In addition the Media Presentation Description may be updated in dynamic Media Presentations, i.e. the `@minimumUpdatePeriod` may be present. NOTE Static Media Presentations are typically used for On-Demand services, whereas dynamic Media Presentations are used for live services.
`@availabilityStartTime`	CM Must be present for `@type='dynamic'`	For `@type='dynamic'` this attribute shall be present. In this case it specifies the anchor for the computation of the earliest availability time (in UTC) for any Segment in the Media Presentation. For `@type="static"` if present, it specifies the Segment availability start time for all Segments referred to in this MPD. If not present, all Segments described in the MPD shall become available at the time the MPD becomes available.
`@publishTime`	OD Must be present for `@type='dynamic'`	specifies the wall-clock time when the MPD was generated and published at the origin server. MPDs with a later value of `@publishTime` shall be an update as defined in 5.4 to MPDs with earlier `@publishTime`.

Element or Attribute Name	Use	Description
@availabilityEndTime	O	specifies the latest Segment availability end time for any Segment in the Media Presentation. When not present, the value is unknown.
@mediaPresentationDuration	O	specifies the duration of the entire Media Presentation. If the attribute is not present, the duration of the Media Presentation is unknown. This attribute shall be present when neither the attribute **MPD**@minimumUpdatePeriod nor the **Period**@duration of the last Period are present.
@minimumUpdatePeriod	O	If this attribute is present, it specifies the smallest period between potential changes to the MPD. This can be useful to control the frequency at which a client checks for updates. If this attribute is not present it indicates that the MPD does not change. If **MPD**@type is 'static', @minimumUpdatePeriod shall not be present. Details on the use of the value of this attribute are specified in 5.4.
@minBufferTime	M	specifies a common duration used in the definition of the Representation data rate (see @bandwidth attribute in 5.3.5.2).
@timeShiftBufferDepth	O	specifies the duration of the smallest time shifting buffer for any Representation in the MPD that is guaranteed to be available for a Media Presentation with type 'dynamic'. When not present, the value is infinite. This value of the attribute is undefined if the type attribute is equal to 'static'.
@suggestedPresentationDelay	O	when @type is 'dynamic', it specifies a fixed delay offset in time from the presentation time of each access unit that is suggested to be used for presentation of each access unit. For more details refer to 7.2.1. When not specified, then no value is provided and the client is expected to choose a suitable value. when @type is 'static' the value of the attribute is undefined and may be ignored.
@maxSegmentDuration	O	specifies the maximum duration of any Segment in any Representation in the Media Presentation, i.e. documented in this MPD and any future update of the MPD. If not present, then the maximum Segment duration shall be the maximum duration of any Segment documented in this MPD.
@maxSubsegmentDuration	O	specifies the maximum duration of any Media Subsegment in any Representation in the Media Presentation. If not present, the same value as for the maximum Segment duration is implied.

Element or Attribute Name	Use	Description
ProgramInformation	0…N	specifies descriptive information about the program. For more details refer to the description in 5.7.
BaseURL	0…N	specifies a Base URL that can be used for reference resolution and alternative URL selection. For more details refer to the description in 5.6.
Location	0…N	specifies a location at which the MPD is available.
Period	1…N	specifies the information of a Period. For more details refer to the description in 5.3.2.
Metrics	0 ... N	specifies the DASH Metrics. For more details see 5.9.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>…<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @		

5.3.1.3 XML syntax

```
  <!-- MPD Type -->
  <xs:complexType name="MPDtype">
    <xs:sequence>
      <xs:element name="ProgramInformation" type="ProgramInformationType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="BaseURL" type="BaseURLType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Location" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Period" type="PeriodType" maxOccurs="unbounded"/>
      <xs:element name="Metrics" type="MetricsType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="xs:string"/>
    <xs:attribute name="profiles" type="xs:string" use="required"/>
    <xs:attribute name="type" type="PresentationType" default="static"/>
    <xs:attribute name="availabilityStartTime" type="xs:dateTime"/>
    <xs:attribute name="availabilityEndTime" type="xs:dateTime"/>
    <xs:attribute name="publishTime" type="xs:dateTime"/>
    <xs:attribute name="mediaPresentationDuration" type="xs:duration"/>
    <xs:attribute name="minimumUpdatePeriod" type="xs:duration"/>
    <xs:attribute name="minBufferTime" type="xs:duration" use="required"/>
    <xs:attribute name="timeShiftBufferDepth" type="xs:duration"/>
    <xs:attribute name="suggestedPresentationDelay" type="xs:duration"/>
    <xs:attribute name="maxSegmentDuration" type="xs:duration"/>
    <xs:attribute name="maxSubsegmentDuration" type="xs:duration"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Presentation Type enumeration -->
  <xs:simpleType name="PresentationType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="static"/>
      <xs:enumeration value="dynamic"/>
    </xs:restriction>
  </xs:simpleType>
```

5.3.2 Period

5.3.2.1 Overview

A Media Presentation consists of one or more Periods. A Period is defined by a **Period** element in the **MPD** element.

The type of the Period, either a regular Period or an Early Available Period, as well as the *PeriodStart* time of a regular Period is determined as follows:

- If the attribute `@start` is present in the **Period**, then the Period is a regular Period and the *PeriodStart* is equal to the value of this attribute.
- If the `@start` attribute is absent, but the previous **Period** element contains a `@duration` attribute then this new Period is also a regular Period. The start time of the new Period *PeriodStart* is the sum of the start time of the previous Period *PeriodStart* and the value of the attribute `@duration` of the previous Period.
- If (i) `@start` attribute is absent, and (ii) the **Period** element is the first in the MPD, and (iii) the **MPD**`@type` is `'static'`, then the *PeriodStart* time shall be set to zero.
- If (i) `@start` attribute is absent, and (ii) the previous **Period** element does not contains a `@duration` attribute or the **Period** element is the first in the MPD, and (iii) the **MPD**`@type` is `'dynamic'`, then this Period is an Early Available Period (see below for details).

For any regular Period the following holds: *PeriodStart* reflects the actual time that should elapse after playing the media of all prior Periods in this Media Presentation relative to the *PeriodStart* time of the first Period in the Media Presentation. The Period extends until the *PeriodStart* of the next Period, or until the end of the Media Presentation in the case of the last Period. More specifically, the difference between the *PeriodStart* time of a Period and either the *PeriodStart* time of the following Period, if this is not the last Period, or the value of the **MPD**`@mediaPresentationDuration` if this is the last one, is the presentation duration in Media Presentation time of the media content represented by the Representations in this Period.

Early Available Periods may be used to advertise initialization of other non-media data before the media data itself is available. **Period** elements documenting early available Periods shall not occur before any **Period** element documenting a regular Period. For Early Available Periods, any resources that are announced in such a **Period** element shall be available. Such a **Period** element shall not contain URLs to Media Segments. The data contained in such a **Period** element does not represent a Period in the Media Presentation. Only when the *PeriodStart* time becomes known through an update of the MPD, such a **Period** element represents a regular Period. However, an update of the MPD may even remove a **Period** element representing an Early Available Period in later updates of the MPD as long as no *PeriodStart* time is associated with the Period.

To avoid dereferencing of a remote element entity containing a **Period** element solely to determine the Period timeline, e.g. in case of seeking, **Period**`@start` or previous Period's **Period**`@duration` should be present in the MPD.

The semantics of the attributes and elements within a **Period** element are provided in Table 4 of 5.3.2.2. The XML syntax of the **Period** element is provided in 5.3.2.3.

5.3.2.2 Semantics

Table 4 — Semantics of `Period` element

Element or Attribute Name	Use	Description
`Period`		specifies the information of a Period.
`@xlink:href`	O	specifies a reference to a remote element entity that is either empty or contains one or multiple top-level elements of type **`Period`**
`@xlink:actuate`	OD default: `onRequest`	specifies the processing instructions, which can be either "`onLoad`" or "`onRequest`". This attribute shall not be present if the `@xlink:href` attribute is not present.
`@id`	O	specifies an identifier for this Period. The identifier shall be unique within the scope of the Media Presentation. If the **`MPD`**`@type` is "`dynamic`", then this attribute shall be present and shall not change in case the MPD is updated. If not present, no identifier for the Period is provided.
`@start`	O	if present, specifies the *PeriodStart* time of the Period. The *PeriodStart* time is used as an anchor to determine the MPD start time of each Media Segment as well as to determine the presentation time of each access unit in the Media Presentation timeline. If not present, refer to the details in 5.3.2.1.
`@duration`	O	if present specifies the duration of the Period to determine the *PeriodStart* time of the next Period. If not present, refer to the details in 5.3.2.1.
`@bitstreamSwitching`	OD Default: `false`	When set to '`true`', this is equivalent as if the **`AdaptationSet`**`@bitstreamSwitching` for each Adaptation Set contained in this Period is set to '`true`'. In this case, the **`AdaptationSet`**`@bitstreamSwitching` attribute shall not be set to '`false`' for any Adaptation Set in this Period.
`BaseURL`	0…N	specifies a base URL that can be used for reference resolution and alternative URL selection. For more details refer to the description in 5.6.
`SegmentBase`	0...1	specifies default Segment Base information. Information in this element is overridden by information in **`AdapationSet.SegmentBase`** and **`Representation.SegmentBase`**, if present. For more details see 5.3.9.

Element or Attribute Name	Use	Description
SegmentList	0...1	specifies default Segment List information. Information in this element is overridden by information in **AdapationSet.SegmentList** and **Representation.SegmentList**, if present. For more details see 5.3.9.
SegmentTemplate	0...1	specifies default Segment Template information. Information in this element is overridden by information in **AdapationSet.SegmentTemplate** and **Representation.SegmentTemplate**, if present. For more details see 5.3.9.
AssetIdentifier	0...1	specifies that this Period belongs to a certain asset. For more details see 5.8.5.7.
EventStream	0...N	specifies an event stream. For more details see 5.10.2.
AdaptationSet	0...N	specifies an Adaptation Set. At least one Adaptation Set shall be present in each Period unless the value of the `@duration` attribute of the Period is set to zero. For more details see 5.3.3.
Subset	0...N	specifies a Subset. For more details see 5.3.8.

Legend:
For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory.
For elements: <minOccurs>...<maxOccurs> (N=unbounded)
Note that the conditions only holds without using `xlink:href`. If linking is used, then all attributes are "optional" and <minOccurs=0>
Elements are **bold**; attributes are non-bold and preceded with an @.

5.3.2.3 XML syntax

```
  <!-- Period -->
  <xs:complexType name="PeriodType">
    <xs:sequence>
      <xs:element name="BaseURL" type="BaseURLType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="SegmentBase" type="SegmentBaseType" minOccurs="0"/>
      <xs:element name="SegmentList" type="SegmentListType" minOccurs="0"/>
      <xs:element name="SegmentTemplate" type="SegmentTemplateType" minOccurs="0"/>
      <xs:element name="AssetIdentifier" type="DescriptorType" minOccurs="0"/>
      <xs:element name="EventStream" type="EventStreamType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="AdaptationSet" type="AdaptationSetType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="Subset" type="SubsetType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="xlink:href"/>
    <xs:attribute ref="xlink:actuate" default="onRequest"/>
    <xs:attribute name="id" type="xs:string" />
    <xs:attribute name="start" type="xs:duration"/>
    <xs:attribute name="duration" type="xs:duration"/>
    <xs:attribute name="bitstreamSwitching" type="xs:boolean" default="false"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
```

5.3.3 Adaptation Sets

5.3.3.1 Overview

Each Period consists of one or more Adaptation Sets. An Adaptation Set is described by an **`AdaptationSet`** element. **`AdaptationSet`** elements are contained in a **`Period`** element.

An Adaptation Set contains alternate Representations, i.e. only one Representation within an Adaptation Set is expected to be presented at a time. All Representations contained in one Adaptation Set represent the same media content components and therefore contain media streams that are considered to be perceptually equivalent. The Adaptation Set and the contained Representations shall be prepared and contain sufficient information such that seamless switching across different Representations in one Adaptation Set is enabled. If an Adaptation Set is expected to be consumed by DASH clients with restrictions in terms of switching, then the Media Presentation author should provide sufficient means to enable seamless switching under these restrictions.

Representations are arranged into Adaptation Sets according to the media content component properties of the media content components present in the Representations, namely

- the language as described by the `@lang` attribute,
- the media component type described by the `@contentType` attribute,
- the picture aspect ratio as described by the `@par` attribute,
- the role property as described by the **`Role`** elements,
- the accessibility property as described by the **`Accessibility`** elements,
- the viewpoint property as described by the **`Viewpoint`** elements,
- the rating property as described by the **`Rating`** elements.

Representations shall appear in the same Adaptation Set if and only if they have identical values for all of these media content component properties for each media content component.

The values for the elements **Role**, **Accessibility**, **Viewpoint** and **Rating** are generally not provided within the scope of this part of ISO/IEC 23009. However, a number of simple schemes are defined in 5.8.5.

If there exist multiple media content components then the properties of each media content component shall be described by a separate **ContentComponent** element as defined in 5.3.4. The **ContentComponent** element shares common elements and attributes with the **AdaptationSet** element. Default values, or values applicable to all media content components, may be provided directly in the **AdaptationSet** element. Attributes present in the **AdaptationSet** shall not be repeated in the **ContentComponent** element.

The **AdaptationSet** element may contain default values for elements and attributes associated to the contained Representations. The list of possible present elements and attributes that are common to **AdaptationSet** and **Representation** (and also **SubRepresentation)** are collected in 5.3.7. Any of the common attributes shall only be present either in the **AdaptationSet** element or in the **Representation** element, but not in both.

The **AdaptationSet** element also supports the description of ranges for the `@bandwidth`, `@width`, `@height` and `@frameRate` attributes associated to the contained Representations, which provide a summary of all values for all the Representations within this Adaptation Set. The Representations contained within an Adaptation Set shall not contain values outside the ranges documented for that Adaptation Set.

Adaptation Sets may be further arranged into groups using the `@group` attribute. The semantics of this grouping is that the media content within one Period is represented by:

1) either one Representation from group 0, if present,

2) or the combination of at most one Representation from each non-zero group.

If the **AdaptationSet**`@group` attribute is not present then all Representations in this Adaptation Set are assigned to a non-zero group specific to this Adaptation Set.

The semantics of the attributes and elements within an **AdaptationSet** element are provided in Table 5 of 5.3.3.2. The XML syntax of the **AdaptationSet** element is provided in 5.3.3.3.

5.3.3.2 Semantics

Table 5 — Semantics of `AdaptationSet` element

Element or Attribute Name	Use	Description
AdaptationSet		Adaptation Set description
`@xlink:href`	O	specifies a reference to a remote element entity that shall contain exactly one element of type **AdaptationSet**
`@xlink:actuate`	OD default: '`onRequest`'	specifies the processing instructions, which can be either "`onLoad`" or "`onRequest`".
`@id`	O	specifies an unique identifier for this Adaptation Set in the scope of the Period. The attribute shall be unique in the scope of the containing Period. The attribute shall not be present in a remote element entity.

Element or Attribute Name	Use	Description
		If not present, no identifier for the Adaptation Set is specified.
@group	O	specifies an identifier for the group that is unique in the scope of the containing Period. For details refer to 5.3.3.1.
CommonAttributesElements	-	specifies the common attributes and elements (attributes and elements from base type ***RepresentationBaseType***). For details see 5.3.7.
@lang	O	Declares the language code for this Adaptation Set. The syntax and semantics according to IETF RFC 5646 shall be used. If not present, the language code may be defined for each media component or it may be unknown.
@contentType	O	specifies the media content component type for this Adaptation Set. A value of the top-level Content-type 'type' value as defined in RFC4288, Clause 4 shall be taken. If not present, the media content component type may be defined for each media component or it may be unknown.
@par	O	specifies the picture aspect ratio of the video media component type, in the form of a string consisting of two integers separated by ':', e.g.,"16:9". When this attribute is present, and the attributes @width and @height for the set of Representations are also present, the picture aspect ratio as specified by this attribute shall be the same as indicated by the values of @width, @height, and @sar, i.e. it shall express the same ratio as (@width * *sarx*) : (@height * *sary*), with *sarx* the first number in @sar and *sary* the second number. If not present, the picture aspect ratio may be defined for each media component or it may be unknown.
@minBandwidth	O	specifies the minimum @bandwidth value in all Representations in this Adaptation Set. This value has the same units as the @bandwidth attribute. If not present, the value is unknown.
@maxBandwidth	O	specifies the maximum @bandwidth value in all Representations in this Adaptation Set. This value has the same units as the @bandwidth attribute. If not present, the value is unknown.
@minWidth	O	specifies the minimum @width value in all Representations in this Adaptation Set. This value has the same units as the @width attribute.

Element or Attribute Name	Use	Description
		If not present, the value is unknown.
@maxWidth	O	specifies the maximum @width value in all Representations in this Adaptation Set. This value has the same units as the @width attribute. If not present, the value is unknown.
@minHeight	O	specifies the minimum @height value in all Representations in this Adaptation Set. This value has the same units as the @height attribute. If not present, the value is unknown.
@maxHeight	O	specifies the maximum @height value in all Representations in this Adaptation Set. This value has the same units as the @height attribute. If not present, the value is unknown.
@minFrameRate	O	specifies the minimum @framerate value in all Representations in this Adaptation Set. This value is encoded in the same format as the @frameRate attribute. If not present, the value is unknown.
@maxFrameRate	O	specifies the maximum @framerate value in all Representations in this Adaptation Set. This value is encoded in the same format as the @frameRate attribute. If not present, the value is unknown.
@segmentAlignment	OD default: false	when not set to 'false', this specifies that for any two Representations, X and Y, within the same Adaptation Set, the *m*-th Segment of X and the *n*-th Segment of Y are non-overlapping (as defined in 4.5.3) whenever *m* is not equal to *n*. For Adaptation Sets containing Representations with multiple media content components, this attribute value shall be either 'true' or 'false'. For Adaptation Sets containing Representations with a single media content component, when two **AdaptationSet** elements within a Period share the same integer value for this attribute, then for any two Representations, X and Y, within the union of the two Adaptation Sets, the *m*-th Segment of X and the *n*-th Segment of Y are non-overlapping (as defined in 4.5.3) whenever *m* is not equal to *n*.
@bitstreamSwitching	O	When this flag is set to 'true', the following applies: • All Representations in the Adaptation Set shall have the same number *M* of Media Segments; • Let $R_1, R_2, ..., R_N$ be all the Representations within

Element or Attribute Name	Use	Description
		the Adaptation Set. • Let ○ $S_{i,j}$, for $j > 0$, be the j^{th} Media Segment in the i^{th} Representation (i.e., R_i) ○ if present, let $S_{i,0}$ be the Initialization Segment in the i^{th} Representation, and ○ if present, let B_i be the Bitstream Switching Segment in the i^{th} Representation. • The sequence of ○ any Initialization Segment, if present, in the Adaptation Set, with, ○ if Bitstream Switching Segments are present, $B_{i(1)}, S_{i(1),1}, B_{i(2)}, S_{i(2),2}, \ldots, B_{i(k)}, S_{i(k),k}, \ldots, B_{i(M)}, S_{i(M),M}$ ○ else $S_{i(1),1}, S_{i(2),2}, \ldots, S_{i(k),k}, \ldots, S_{i(M),M}$, wherein any $i(k)$ for all k values in the range of 1 to M, respectively, is an integer value in the range of 1 to N, results in a "conforming Segment sequence" as defined in 4.5.4 with the media format as specified in the `@mimeType` attribute. More detailed rules may be defined for specific media formats.
`@subsegmentAlignment`	OD default: `false`	If the `@subsegmentAlignment` for an Adaptation Set is set to other than '`false`', all following conditions shall be satisfied: — Each Media Segment shall be indexed (i.e. either it contains a Segment index or there is an Index Segment providing an index for the Media Segment) — For any two Representations, X and Y, within the same Adaptation Set, the m-th Subsegment of X and the n-th Subsegment of Y are non-overlapping (as defined in 4.5.3) whenever m is not equal to n. — For Adaptation Sets containing Representations with a single media content component, when two **`AdaptationSet`** elements within a Period share

Element or Attribute Name	Use	Description
		the same integer value for this attribute, then for any two Representations, X and Y, within the union of the two Adaptation Sets, the *m*-th Subsegment of X and the *n*-th Subsegment of Y are non-overlapping (as defined in 4.5.3) whenever *m* is not equal to *n*.
`@subsegmentStartsWithSAP`	OD default: `0`	when greater than 0, specifies that each Subsegment with `SAP_type` greater than 0 starts with a SAP of type less than or equal to the value of `@subsegmentStartsWithSAP`. A Subsegment starts with SAP when the Subsegment contains a SAP, and for the first SAP, I_{SAU} is the index of the first access unit that follows I_{SAP}, and I_{SAP} is contained in the Subsegment. The semantics of `@subsegmentStartsWithSAP` equal to 0 are unspecified.
`Accessibility`	0 … N	specifies information about accessibility scheme For more details refer to 5.8.1 and 5.8.4.3.
`Role`	0 … N	specifies information on role annotation scheme For more details refer to 5.8.1 and 5.8.4.2.
`Rating`	0 … N	specifies information on rating scheme. For more details refer to 5.8.1 and 5.8.4.4.
`Viewpoint`	0 … N	specifies information on viewpoint annotation scheme. For more details refer to 5.8.1 and 5.8.4.5.
`ContentComponent`	0…N	specifies the properties of one media content component contained in this Adaptation Set. For more details refer to 5.3.4.
`BaseURL`	0…N	specifies a base URL that can be used for reference resolution and alternative URL selection. For more details refer to the description in 5.6.
`SegmentBase`	0...1	specifies default Segment Base information. Information in this element is overridden by information in the **`Representation.SegmentBase`**, if present. For more details see 5.3.9.
`SegmentList`	0...1	specifies default Segment List information. Information in this element is overridden by information in the **`Representation.SegmentList`**, if present.

Element or Attribute Name	Use	Description
		For more details see 5.3.9.
SegmentTemplate	0...1	specifies default Segment Template information. Information in this element is overridden by information in the **Representation.SegmentTemplate**, if present. For more details see 5.3.9.
Representation	0 … N	specifies a Representation. At least one Representation element shall be present in each Adaptation Set. The actual element may however be part of a remote element entity if xlink is used on the containing **AdaptationSet** element. For more details refer to 5.3.5.

Legend:
For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory, F=Fixed.
For elements: <minOccurs>...<maxOccurs> (N=unbounded)
Note that the conditions only holds without using `xlink:href`. If linking is used, then all attributes are "optional" and <minOccurs=0>
Elements are **bold**; attributes are non-bold and preceded with an @, List of elements and attributes is in ***italics bold*** referring to those taken from the Base type that has been extended by this type.

5.3.3.3 XML syntax

```
  <!-- Adaptation Set -->
  <xs:complexType name="AdaptationSetType">
    <xs:complexContent>
      <xs:extension base="RepresentationBaseType">
        <xs:sequence>
          <xs:element name="Accessibility" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
          <xs:element name="Role" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="Rating" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="Viewpoint" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="ContentComponent" type="ContentComponentType" minOccurs="0"
maxOccurs="unbounded"/>
          <xs:element name="BaseURL" type="BaseURLType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="SegmentBase" type="SegmentBaseType" minOccurs="0"/>
          <xs:element name="SegmentList" type="SegmentListType" minOccurs="0"/>
          <xs:element name="SegmentTemplate" type="SegmentTemplateType" minOccurs="0"/>
          <xs:element name="Representation" type="RepresentationType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute ref="xlink:href"/>
        <xs:attribute ref="xlink:actuate" default="onRequest"/>
        <xs:attribute name="id" type="xs:unsignedInt"/>
        <xs:attribute name="group" type="xs:unsignedInt"/>
        <xs:attribute name="lang" type="xs:language"/>
        <xs:attribute name="contentType" type="xs:string"/>
        <xs:attribute name="par" type="RatioType"/>
        <xs:attribute name="minBandwidth" type="xs:unsignedInt"/>
        <xs:attribute name="maxBandwidth" type="xs:unsignedInt"/>
        <xs:attribute name="minWidth" type="xs:unsignedInt"/>
        <xs:attribute name="maxWidth" type="xs:unsignedInt"/>
        <xs:attribute name="minHeight" type="xs:unsignedInt"/>
        <xs:attribute name="maxHeight" type="xs:unsignedInt"/>
        <xs:attribute name="minFrameRate" type="FrameRateType"/>
        <xs:attribute name="maxFrameRate" type="FrameRateType"/>
        <xs:attribute name="segmentAlignment" type="ConditionalUintType" default="false"/>
        <xs:attribute name="subsegmentAlignment" type="ConditionalUintType" default="false"/>
        <xs:attribute name="subsegmentStartsWithSAP" type="SAPType" default="0"/>
        <xs:attribute name="bitstreamSwitching" type="xs:boolean"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <!-- Ratio Type for sar and par -->
  <xs:simpleType name="RatioType">
    <xs:restriction base="xs:string">
      <xs:pattern value="[0-9]*:[0-9]*"/>
    </xs:restriction>
  </xs:simpleType>

  <!-- Type for Frame Rate -->
  <xs:simpleType name="FrameRateType">
    <xs:restriction base="xs:string">
      <xs:pattern value="[0-9]*[0-9](/[0-9]*[0-9])?"/>
    </xs:restriction>
  </xs:simpleType>


  <!-- Conditional Unsigned Integer (unsignedInt or boolean) -->
  <xs:simpleType name="ConditionalUintType">
    <xs:union memberTypes="xs:unsignedInt xs:boolean"/>
  </xs:simpleType>
```

5.3.4 Media Content Component

5.3.4.1 Overview

Each Adaptation Set contains one or more media content components. The properties of each media content component are described by a **`ContentComponent`** element or may be described directly on the **`AdaptationSet`** element if only one media content component is present in the Adaptation Set. **`ContentComponent`** elements are contained in an **`AdaptationSet`** element.

The semantics of the attributes and elements within a **ContentComponent** element are provided in Table 6 of 5.3.4.2. The XML syntax of the **ContentComponent** element is provided in 5.3.4.3.

5.3.4.2 Semantics

Table 6 — Semantics of ContentComponent element

Element or Attribute Name	Use	Description
ContentComponent		description of a content component
@id	O	specifies an identifier for this media component. The attribute shall be unique in the scope of the containing Adaptation Set.
@lang	O	same semantics as in Table 5 for @lang attribute
@contentType	O	same semantics as in Table 5 for @contentType attribute
@par	O	same semantics as in Table 5 for @par attribute.
Accessibility	0 … N	same semantics as in Table 5 for **Accessibility** element
Role	0 … N	same semantics as in Table 5 for **Role** element
Rating	0 … N	same semantics as in Table 5 for **Rating** element
Viewpoint	0 … N	same semantics as in Table 5 for **Viewpoint** element
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory, F=Fixed. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @, List of elements and attributes is in ***italics bold*** referring to those taken from the Base type that has been extended by this type.		

5.3.4.3 XML syntax

```
<!-- Content Component -->
<xs:complexType name="ContentComponentType">
  <xs:sequence>
    <xs:element name="Accessibility" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Role" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Rating" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="Viewpoint" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:unsignedInt"/>
  <xs:attribute name="lang" type="xs:language"/>
  <xs:attribute name="contentType" type="xs:string"/>
  <xs:attribute name="par" type="RatioType"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

5.3.5 Representation

5.3.5.1 Overview

Representations are described by the **`Representation`** element. **`Representation`** elements are contained in an **`AdaptationSet`** element.

A Representation is one of the alternative choices of the complete set or subset of media content components comprising the media content during the defined Period.

A Representation starts at the start of the Period *PeriodStart* and continues to the end of the Period, i.e. the start of the next Period or the end of the Media Presentation.

Each Representation includes one or more media streams, where each media stream is an encoded version of one media content component.

A Representation consists of one or more Segments.

Each Representation either shall contain an Initialization Segment or each Media Segment in the Representation shall be self-initializing, i.e. the Media Segment itself conforms to the media type as specified in the `@mimeType` attribute for this Representation.

When a Representation is not a dependent Representation, i.e. the `@dependencyId` attribute is absent, then concatenation of the Initialization Segment, if present, and all consecutive Media Segments in one Representation shall represent a conforming Segment sequence as defined in 4.5.4 conforming to the media type as specified in the `@mimeType` attribute for this Representation.

Dependent Representations are described by a **`Representation`** element that contains a `@dependencyId` attribute. Dependent Representations are regular Representations except that they depend on a set of complementary Representations for decoding and/or presentation. The `@dependencyId` contains the values of the `@id` attribute of all the complementary Representations, i.e. Representations that are necessary to present and/or decode the media content components contained in this dependent Representation.

For any dependent Representation X that depends on complementary Representation Y, the *m*-th Subsegment of X and the *n*-th Subsegment of Y shall be non-overlapping (as defined in 4.5.3) whenever *m* is not equal to *n*. For dependent Representations the concatenation of the Initialization Segment with the sequence of Subsegments of the dependent Representations, each being preceded by the corresponding Subsegment of each of the complementary Representations in order as provided in the `@dependencyId` attribute shall represent a conforming Subsegment sequence as defined in 4.5.4 conforming to the media format as specified in the `@mimeType` attribute for this dependent Representation.

NOTE When decoding of a dependent Representation is started from a SAP in the (Sub)Segment with number *i*, the decoding process does not need to access data from the complementary Representation(s) from any earlier (sub)segments than (sub)Segment with number *i* of the complementary Representation(s).

If a Representation is offered in a Media Presentation with **`MPD`**`@type='dynamic'`, it is recommended that means to compensate such drift be included. For more details refer to A.8.

The semantics of the attributes and elements within a Representation are provided in Table 7 of 5.3.5.2. The XML syntax of the Representation type is provided in 5.3.5.3.

5.3.5.2 Semantics

Table 7 —Semantics of `Representation` element

Element or Attribute Name	Use	Description
`Representation`		This element contains a description of a Representation.
`@id`	M	specifies an identifier for this Representation. The identifier shall be unique within a Period unless the Representation is functionally identically to another Representation in the same Period. The identifier shall not contain whitespace characters. If used in the template-based URL construction as defined in 5.3.9.4.4, the string shall only contain characters that are permitted within an HTTP-URL according to RFC 3986.
`@bandwidth`	M	Consider a hypothetical constant bitrate channel of bandwidth with the value of this attribute in bits per second (bps). Then, if the Representation is continuously delivered at this bitrate, starting at any SAP that is indicated either by `@startWithSAP` or by any Segment Index box, a client can be assured of having enough data for continuous playout providing playout begins after `@minBufferTime` * `@bandwidth` bits have been received (i.e. at time `@minBufferTime` after the first bit is received). For dependent Representations this value specifies the bandwidth according to the above definition for the aggregation of this Representation and all complementary Representations.
`@qualityRanking`	O	specifies a quality ranking of the Representation relative to other Representations in the same Adaptation Set. Lower values represent higher quality content. If not present then no ranking is defined.
`@dependencyId`	O	specifies all complementary Representations the Representation depends on in the decoding and/or presentation process as a whitespace-separated list of values of `@id` attributes. If not present, the Representation can be decoded and presented independently of any other Representation. This attribute shall not be present where there are no dependencies.
`@mediaStreamStructureId`	O	The attribute may be present for Representations containing video and its semantics are unspecified for any other type of Representations. If present, the attribute `@mediaStreamStructureId` specifies a whitespace-separated list of media stream structure identifier values. If media streams share the same media stream structure identifier value, the media streams

Element or Attribute Name	Use	Description
		shall have the following characteristics: - The media streams have the same number of Stream Access Points of type 1 to 3. - The values of T_{SAP}, T_{DEC}, T_{EPT}, and T_{PTF} of the *i*-th SAP of type 1 to 3 in one media stream are identical to the values of T_{SAP}, T_{DEC}, T_{EPT}, and T_{PTF}, respectively, of the *i*-th SAP of type 1 to 3 in the other media streams for any value of *i* from 1 to the number of SAPs of type 1 to 3 in any of the media streams. - A media stream formed by concatenating the media stream of a first Representation until I_{SAU} (exclusive) of the *i*-th SAP of type 1 to 3 and the media stream of a second Representation (having the same media stream structure identifier value as for the first Representation) starting from the I_{SAU} (inclusive) of the *i*-th SAP of type 1 to 3 conforms to the specification in which the media stream format is specified for any value of *i* from 1 to the number of SAPs of type 1 to 3 in either media stream. Furthermore, the decoded pictures have an acceptable quality regardless of type of the Stream Access Point access unit used. All media stream structure identifier values for one Adaptation Set shall differ from those of another Adaptation Set. If not present, then for this Representation no similarities to other Representations are known. NOTE Indicating multiple media stream structure identifier values for a Representation can be useful in cases where switching between Representations A and B as well as between Representations B and C is allowed at non-IDR intra pictures, but switching between Representations A and C would cause too severe a degradation in the quality of the leading pictures and is hence not allowed. To indicate these permissions and restrictions, Representation A would contain `@mediaStreamStructureId` equal to “1”, Representation B would contain `@mediaStreamStructureId` equal to “1 2”, and Representation C would contain `@mediaStreamStructureId` equal to “2”
CommonAttributesElements	-	Common Attributes and Elements (attributes and elements from base type ***RepresentationBaseType***), for more details see 5.3.7
BaseURL	0…N	specifies a Base URL that can be used for reference resolution and alternative URL selection. For more details refer to the description in 5.6.

Element or Attribute Name	Use	Description
SubRepresentation	0 … N	specifies information about a Sub-Representation that is embedded in the containing Representation. For more details see 5.3.6.
SegmentBase	0...1	specifies default Segment Base information. For more details see 5.3.9.
SegmentList	0 ... 1	specifies the Segment List information. For more details see 5.3.9.
SegmentTemplate	0 ... 1	specifies the Segment Template information. For more details see 5.3.9.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @, List of elements and attributes is in ***italics bold*** referring to those taken from the Base type that has been extended by this type.		

5.3.5.3 XML syntax

```
  <!-- Representation -->
  <xs:complexType name="RepresentationType">
    <xs:complexContent>
      <xs:extension base="RepresentationBaseType">
        <xs:sequence>
          <xs:element name="BaseURL" type="BaseURLType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="SubRepresentation" type="SubRepresentationType" minOccurs="0"
maxOccurs="unbounded"/>
          <xs:element name="SegmentBase" type="SegmentBaseType" minOccurs="0"/>
          <xs:element name="SegmentList" type="SegmentListType" minOccurs="0"/>
          <xs:element name="SegmentTemplate" type="SegmentTemplateType" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="id" type="StringNoWhitespaceType" use="required"/>
        <xs:attribute name="bandwidth" type="xs:unsignedInt" use="required"/>
        <xs:attribute name="qualityRanking" type="xs:unsignedInt"/>
        <xs:attribute name="dependencyId" type="StringVectorType"/>
        <xs:attribute name="mediaStreamStructureId" type="StringVectorType"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <!-- String without white spaces -->
  <xs:simpleType name="StringNoWhitespaceType">
    <xs:restriction base="xs:string">
      <xs:pattern value="[^\r\n\t \p{Z}]*"/>
    </xs:restriction>
  </xs:simpleType>

  <!-- Whitespace-separated list of strings -->
  <xs:simpleType name="StringVectorType">
    <xs:list itemType="xs:string"/>
  </xs:simpleType>
```

5.3.6 Sub-Representation

5.3.6.1 Overview

Sub-Representations are embedded in regular Representations and are described by the **SubRepresentation** element. **SubRepresentation** elements are contained in a **Representation** element.

The **SubRepresentation** element describes properties of one or several media content components that are embedded in the Representation. It may for example describe the exact properties of an embedded audio component (e.g., codec, sampling rate, etc.), an embedded sub-title (e.g., codec) or it may describe some embedded lower quality video layer (e.g. some lower frame rate, etc.).

Sub-Representations and Representation share some common attributes and elements.

In case the `@level` attribute is present in the **SubRepresentation** element,

- Sub-Representations provide the ability for accessing a lower quality version of the Representation in which they are contained. In this case, Sub-Representations for example allow extracting the audio track in a multiplexed Representation or may allow for efficient fast-forward or rewind operations if provided with lower frame rate;
- the Initialization Segment and/or the Media Segments and/or the Index Segments shall provide sufficient information such that the data can be easily accessed through HTTP partial GET requests. The details on providing such information shall be defined by the media format in use. For media formats defined in this part of ISO/IEC 23009, the Subsegment Index as defined in 6.3.2.4 shall be used.

If the `@level` attribute is absent, then the **SubRepresentation** element is solely used to provide a more detailed description for media streams that are embedded in the Representation.

The semantics of the attributes and elements within a Sub-Representation are provided in Table 8 of 5.3.6.2. The XML syntax of the Sub-Representation type is provided in 5.3.6.3.

5.3.6.2 Semantics

Table 8 —Semantics of SubRepresentation element

Element or Attribute Name	Use	Description
SubRepresentation		specifies a Sub-Representation.
`@level`	O	specifies the Sub-Representation level. If `@level` attribute is present and for media formats used in this Part of ISO/IEC 23009, a Subsegment Index as defined in 6.3.2.4 shall be available for each Media Segment in the containing Representation. If not present, then the **SubRepresentation** element is solely used to provide a more detailed description for media streams that are embedded in the Representation.
`@dependencyLevel`	O	specifies the set of Sub-Representations within this Representation that this Sub-Representation depends on in the decoding and/or presentation process as a whitespace-separated list of `@level` values. If not present, the Sub-Representation can be decoded and presented independently of any other Representation.

Element or Attribute Name	Use	Description
@bandwidth	CM shall be present if @level is present	Identical to the @bandwidth definition in Representation, but applied to this Sub-Representation. This attribute shall be present if the @level attribute is present.
@contentComponent	O	if present, specifies the set of all media content components that are contained in this Sub-Representation as a whitespace-separated list of values of **ContentComponent**@id values. if not present, the Sub-Representation is not assigned to a media content component.
CommonAttributesElements	-	Common Attributes and Elements (attributes and elements from base type ***RepresentationBaseType***). For details see 5.3.7.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold preceded with an @, List of elements and attributes is in ***italics bold*** referring to those taken from the Base type that has been extended by this type.		

5.3.6.3 XML syntax

```
<!-- SubRepresentation -->
<xs:complexType name="SubRepresentationType">
  <xs:complexContent>
    <xs:extension base="RepresentationBaseType">
      <xs:attribute name="level" type="xs:unsignedInt"/>
      <xs:attribute name="dependencyLevel" type="UIntVectorType"/>
      <xs:attribute name="bandwidth" type="xs:unsignedInt"/>
      <xs:attribute name="contentComponent" type="StringVectorType"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- Whitespace-separated list of unsigned integers -->
<xs:simpleType name="UIntVectorType">
  <xs:list itemType="xs:unsignedInt"/>
</xs:simpleType>
```

5.3.7 Common attributes and elements

5.3.7.1 Overview

The elements **AdaptationSet**, **Representation** and **SubRepresentation** have assigned common attributes and elements. The attributes and elements listed in Table 9 of 5.3.7.2 may be present in all three elements.

The semantics of the common attributes and elements are provided in Table 9 in 5.3.7.2, the syntax is provided in 5.3.7.3.

The 'Use' column in Table 9 shall be interpreted that an attribute marked with 'M' shall be available for a Representation, i.e. it shall either be present in the **Representation** element, or if not, it shall be in the containing **AdaptationSet** element. An attribute marked with 'O' may be absent in both.

5.3.7.2 Semantics

Table 9 — Common Adaptation Set, Representation and Sub-Representation attributes and elements

Element or Attribute Name	Use	Description
Common attributes and elements		
`@profiles`	O	specifies the profiles which the associated Representation(s) conform to of the list of Media Presentation profiles as described in 8. The value shall be a subset of the respective value in any higher level of the document hierarchy (Representation, Adaptation Set, MPD). If not present, the value is inferred to be the same as in the next higher level of the document hierarchy. For example, if the value is not present for a Representation, then `@profiles` at the Adaptation Set level is valid for the Representation. The same syntax as defined in 5.3.1.2 shall be used.
`@width`	O	specifies the horizontal visual presentation size of the video media type on a grid determined by the `@sar` attribute. In the absence of `@sar` width and height are specified as if the value of `@sar` were "`1:1`" NOTE The visual presentation size of the video is equal to the number of horizontal and vertical samples used for presentation after encoded samples are cropped in response to encoded cropping parameters, “overscan” signaling, or “pan/scan” display parameters, e.g. SEI messages. If not present on any level, the value is unknown.
`@height`	O	specifies the vertical visual presentation size of the video media type, on a grid determined by the `@sar` attribute. If not present on any level, the value is unknown.
`@sar`	O	specifies the sample aspect ratio of the video media component type, in the form of a string consisting of two integers separated by ‘:’, e.g.,”`10:11`”. The first number specifies the horizontal size of the encoded video pixels (samples) in arbitrary units. The second number specifies the vertical size of the encoded video pixels (samples) in same units as the horizontal size. If not present on any level, the value is unknown.
`@frameRate`	O	specifies the output frame rate (or in the case of interlaced, half the output field rate) of the video media type in the Representation. If the frame or field rate is varying, the value is the average frame or half the average field rate field rate over the entire duration of the Representation. The value is coded as a string, either containing two integers separated by a "/", ("F/D"), or a single integer "F". The frame

Element or Attribute Name	Use	Description
		rate is the division F/D, or F, respectively, per second (i.e. the default value of D is "1"). If not present on any level, the value is unknown.
@audioSamplingRate	O	Either a single decimal integer value specifying the sampling rate or a whitespace separated pair of decimal integer values specifying the minimum and maximum sampling rate of the audio media component type. The values are in samples per second. If not present on any level, the value is unknown.
@mimeType	M	specifies the MIME type of the concatenation of the Initialization Segment, if present, and all consecutive Media Segments in the Representation.
@segmentProfiles	O	specifies the profiles of Segments that are essential to process the Representation. The detailed semantics depend on the value of the @mimeType attribute. The contents of this attribute shall conform to either the pro-simple or pro-fancy productions of RFC6381, Section 4.5, without the enclosing DQUOTE characters, i.e. including only the unencodedv or encodedv elements respectively. As profile identifier the brand identifier for the Segment as defined in 6 shall be used. If not present on any level, the value may be deducted from the value of the @profiles attribute.
@codecs	O	specifies the codecs present within the Representation. The codec parameters shall also include the profile and level information where applicable. For segment formats defined in this specification this element shall be present and the contents of this attribute shall conform to either the simp-list or fancy-list productions of RFC6381, Section 3.2, without the enclosing DQUOTE characters. The codec identifier for the Representation's media format, mapped into the name space for codecs as specified in RFC6381, Section 3.3, shall be used.
@maximumSAPPeriod	O	when present, specifies the maximum SAP interval in seconds of all contained media streams, where the SAP interval is the maximum time interval between the T_{SAP} of any two successive SAPs of types 1 to 3 inclusive of one media stream in the associated Representations. If not present on any level, the value is unknown.
@startWithSAP	O	when present and greater than 0, specifies that in the associated Representations, each Media Segment starts with a SAP of type less than or equal to the value of this attribute value in each media stream.

Element or Attribute Name	Use	Description
		A Media Segment starts with a SAP in a media stream if the stream contains a SAP in that Media Segment, I_{SAU} is the index of the first access unit that follows I_{SAP} and I_{SAP} is contained in the Media Segment. If not present on any level, the value is unknown.
`@maxPlayoutRate`	O	specifies the maximum playout rate as a multiple of the regular playout rate, which is supported with the same decoder profile and level requirements as the normal playout rate. If not present on any level, the value is 1.
`@codingDependency`	O	When present and '`true`', for all contained media streams, specifies that there is at least one access unit that depends on one or more other access units for decoding. When present and '`false`', for any contained media stream, there is no access unit that depends on any other access unit for decoding (e.g. for video all the pictures are intra coded). If not specified on any level, there may or may not be coding dependency between access units.
`@scanType`	O	specifies the scan type of the source material of the video media component type. The value may be equal to one of "progressive", "interlaced" and "unknown". If not specified on any level, the scan type is "`progressive`".
`FramePacking`	0 … N	specifies frame-packing arrangement information of the video media component type. When no **`FramePacking`** element is provided for a video component, frame-packing shall not used for the video media component. For details see 5.8.1 and 5.8.4.6
`AudioChannelConfiguration`	0 … N	specifies the audio channel configuration of the audio media component type. For details see 5.8.1 and 5.8.4.7.
`ContentProtection`	0 … N	specifies information about content protection schemes used for the associated Representations. For details see 5.8.1 and 5.8.4.1.
`EssentialProperty`	0 … N	specifies information about the containing element that is considered essential by the Media Presentation author for processing the containing element. For details see 5.8.4.8.

Element or Attribute Name	Use	Description
SupplementalProperty	0 … N	specifies supplemental information about the containing element that may be used by the DASH client optimizing the processing. For details see 5.8.4.9.
InbandEventStream	0 … N	specifies the presence of an inband event stream in the associated Representations. For details refer to 5.10.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>..<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

5.3.7.3 XML syntax

```
  <!-- Representation base (common attributes and elements) -->
  <xs:complexType name="RepresentationBaseType">
    <xs:sequence>
      <xs:element name="FramePacking" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="AudioChannelConfiguration" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="ContentProtection" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="EssentialProperty" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="SupplementalProperty" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="InbandEventStream" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="profiles" type="xs:string"/>
    <xs:attribute name="width" type="xs:unsignedInt"/>
    <xs:attribute name="height" type="xs:unsignedInt"/>
    <xs:attribute name="sar" type="RatioType"/>
    <xs:attribute name="frameRate" type="FrameRateType"/>
    <xs:attribute name="audioSamplingRate" type="xs:string"/>
    <xs:attribute name="mimeType" type="xs:string"/>
    <xs:attribute name="segmentProfiles" type="xs:string"/>
    <xs:attribute name="codecs" type="xs:string"/>
    <xs:attribute name="maximumSAPPeriod" type="xs:double"/>
    <xs:attribute name="startWithSAP" type="SAPType"/>
    <xs:attribute name="maxPlayoutRate" type="xs:double"/>
    <xs:attribute name="codingDependency" type="xs:boolean"/>
    <xs:attribute name="scanType" type="VideoScanType"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Stream Access Point type enumeration -->
  <xs:simpleType name="SAPType">
    <xs:restriction base="xs:unsignedInt">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="6"/>
    </xs:restriction>
  </xs:simpleType>

  <!-- Video Scan type enumeration -->
  <xs:simpleType name="VideoScanType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="progressive"/>
      <xs:enumeration value="interlaced"/>
      <xs:enumeration value="unknown"/>
    </xs:restriction>
  </xs:simpleType>
```

5.3.8 Subsets

5.3.8.1 Overview

Subsets are described by the **`Subset`** element contained in the **`Period`** element.

Subsets provide a mechanism to restrict the combination of active Adaptation Sets where an active Adaptation Set is one for which the DASH client is presenting at least one of the contained Representations.

A Subset defines a set of one or more Adaptation Sets. The presence of a **`Subset`** element within a **`Period`** element expresses the intention of the creator of the Media Presentation that a client should act as follows: At any time, the set of active Adaptation Sets shall be a subset of the Adaptation Sets of one of the specified Subsets. Any Adaptation Set not explicitly contained in any Subset element is implicitly contained in all specified Subsets.

This implies that

— Empty Subsets are not allowed.

— No Subset should contain all the Adaptation Sets.

Each Adaptation Set for which the value of the `@id` is provided in the `@contains` attribute is contained in this Subset.

The semantics of the attributes and elements within a Subset are provided in Table 10 of 5.3.8.2. The XML syntax of the Subset type is provided in 5.3.8.3.

5.3.8.2 Semantics

Table 10 — `Subset` element semantics

Element or Attribute Name	Use	Description
`Subset`		specifies a Subset
`@contains`	M	specifies the Adaptation Sets contained in a Subset by providing a white-space separated list of the `@id` values of the contained Adaptation Sets.
`@id`	O	specifies a unique identifier for the Subset.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>..<maxOccurs> (N=unbounded) Elements are **`bold`**; attributes are non-bold and preceded with an @.		

5.3.8.3 XML syntax

```
<!-- Subset  -->
<xs:complexType name="SubsetType">
  <xs:attribute name="contains" type="UIntVectorType" use="required"/>
  <xs:attribute name="id" type="xs:string"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

5.3.9 Segments and Segment information

5.3.9.1 General

This clause defines the MPD information for Segments. Segment formats are defined in 6.

Specifically, a Segment shall be referenced by an HTTP-URL included in the MPD, where an HTTP-URL is defined as an `<absolute-URI>` according to RFC 3986, Clause 4.3, with a fixed scheme of “http” or “https”, possibly restricted by a byte range if a range attribute is provided together with the URL. The byte range shall be expressed as a `byte-range-spec` as defined in RFC 2616, Clause 14.35.1. It is restricted to a single expression identifying a contiguous range of bytes.

Each Segment referenced through an HTTP-URL in the MPD is associated with a Segment availability interval, i.e. a time window in wall-clock time at which the Segment can be accessed via the HTTP-URL. The Segment availability interval window is described by a Segment availability start time and a Segment availability end time.

Representations are assigned *Segment Information* through the presence of the elements **BaseURL**, **SegmentBase**, **SegmentTemplate** and/or **SegmentList**. The *Segment Information* provides information on the location, availability and properties of all Segments contained in one Representation. Specifically, information on the presence and location of Initialization, Media, Index and Bitstream Switching Segments is provided.

The elements **SegmentBase**, **SegmentTemplate** and **SegmentList** may be present in the **Representation** element itself. In addition, to express default values, they may be present in the **Period** and **AdaptationSet** element. At each level at most one of the three, **SegmentBase**, **SegmentTemplate** and **SegmentList** shall be present. Further, if **SegmentTemplate** or **SegmentList** is present on one level of the hierarchy, then the other one shall not be present on any lower hierarchy level.

SegmentBase, **SegmentTemplate** and **SegmentList** shall inherit attributes and elements from the same element on a higher level. If the same attribute or element is present on both levels, the one on the lower level shall take precedence over the one on the higher level.

Several mechanisms are available to specify the *Segment Information*. Specifically, each Representation shall have assigned exactly one of the following choices to determine the *Segment Information*, either by direct presence in the **Representation** element or by inheritance from the higher levels:

— one or more **SegmentList** elements - for syntax and semantics refer to 5.3.9.3,

— one **SegmentTemplate** element - for syntax and semantics refer to 5.3.9.4,

— one or more **BaseURL** elements, at most one **SegmentBase** element, and no **SegmentTemplate** or **SegmentList** element. The **SegmentBase** element is defined in 5.3.9.2.

NOTE: These rules do not prohibit the usage of the **BaseURL** element together with **SegmentList** or **SegmentTemplate**. If the BaseURL is present together with the either the **SegmentList** or the **SegmentTemplate**, then processing according to 5.6 is applies.

All three elements **SegmentBase**, **SegmentTemplate** and **SegmentList** share common elements based on the **SegmentBase** element. Furthermore, **SegmentTemplate** and **SegmentList** share common attributes and elements. The common information is defined in 5.3.9.2.

The derivation and details of Initialization, Media, Index and Bitstream Switching Segment Information based on the above information is provided in 5.3.9.5.

5.3.9.2 Segment base information

5.3.9.2.1 Overview

The **SegmentBase** element is sufficient to describe the *Segment Information* if and only if a single Media Segment is provided per Representation and the Media Segment URL is included in the **BaseURL** element.

In case multiple Media Segments are present, either a **SegmentList** or a **SegmentTemplate** shall be used to describe the *Segment Information*. **SegmentList** or a **SegmentTemplate** share the multiple Segment base information as provided in 5.3.9.2.2, Table 12.

If the Representation contains more than one Media Segment, then either the attribute @duration or the element **SegmentTimeline** shall be present. The attribute @duration and the element **SegmentTimeline** shall not be present at the same time.

Segments described by the Segment base information are referenced by an HTTP-URL conforming to the type URLType as defined in Table 13.

The semantics of the attributes and elements for the **SegmentBase** element and the Segment base information are provided in 5.3.9.2.2, Table 11 and the multiple Segment base information in Table 12 in 5.3.9.2.2. The XML syntax of the Segment Base Information is provided in 5.3.9.2.3.

5.3.9.2.2 Semantics

Table 11 — Semantics of SegmentBase element and *Segment Base Information* type

Element or Attribute Name	Use	Description
SegmentBase ***Segment Base Information***		specifies Segment base element. This element also specifies the type for the Segment base information that is the base type for other elements.
@timescale	O	specifies the timescale in units per seconds to be used for the derivation of different real-time duration values in the Segment Information. If not present on any level, it shall be set to 1. NOTE: This may be any frequency but typically is the media clock frequency of one of the media streams (or a positive integer multiple thereof).
@presentationTimeOffset	O	specifies the presentation time offset of the Representation relative to the start of the Period, i.e. the presentation time value of the media stream that shall be presented at the start of this Period. The value of the presentation time offset in seconds is the division of the value of this attribute and the value of the @timescale attribute. If not present on any level, the value of the

		presentation time offset is 0.
@timeShiftBufferDepth	O	specifies the duration of the time shifting buffer for this Representation that is guaranteed to be available for a Media Presentation with type 'dynamic'. When not present, the value is of the @timeShiftBufferDepth on MPD level applies. If present, this value shall be not smaller than the value on MPD level. This value of the attribute is undefined if the type attribute is equal to 'static'. NOTE: When operating in a time-shift buffer on a Representation with value larger than the time-shift buffer than signalled on MPD level, not all Representations may be available for switching.
@indexRange	O	specifies the byte range that contains the Segment Index in all Media Segments of the Representation. The byte range shall be expressed and formatted as a byte-range-spec as defined in RFC 2616, Clause 14.35.1. It is restricted to a single expression identifying a contiguous range of bytes. If not present the value is unknown.
@indexRangeExact	OD default "false"	when set to 'true' specifies that for all Segments in the Representation, the data outside the prefix defined by @indexRange contains the data needed to access all access units of all media streams syntactically and semantically. This attribute shall not be present if @indexRange is absent.
@availabilityTimeOffset	O	specifies an offset to define the adjusted segment availability time. The value is specified in seconds, possibly with arbitrary precision. The offset provides the time how much earlier these segments are available compared to their computed availability start time for all Segments of all associated Representation. The segment availability start time defined by this value is referred to as adjusted segment availability start time. For details on computing the adjusted segment availability start time, refer to 5.3.9.5. If not present, no adjusted segment availability start time is defined. NOTE: the value of "INF" implies availability

		of all segments starts at **MPD**@availabilityStartTime.
@availabilityTimeComplete	O	specifies if all Segments of all associated Representation are complete at the adjusted availability start time. The attribute shall be ignored if @availabilityTimeOffset is not present on any level. If not present on any level, the value is inferred to true. NOTE: If the value is set to false, then it may be inferred by the client that the segment is available at its announced location prior being complete.
Initialization	0 ... 1	specifies the URL including a possible byte range for the Initialization Segment. For the type definition refer to Table 13.
RepresentationIndex	0 ... 1	specifies the URL including a possible byte range for the Representation Index Segment. For the type definition refer to Table 13.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

Table 12 — Semantics of *MultipleSegmentBaseInformation* type

Element or Attribute Name	Use	Description
MultipleSegmentBaseInformation		specifies multiple Segment base information.
@duration	O	If present, specifies the constant approximate Segment duration. All Segments within this Representation element have the same duration unless it is the last Segment within the Period, which could be significantly shorter. The value of the duration in seconds is the division of the value of this attribute and the value of the @timescale attribute associated to the containing Representation. For more details refer to 5.3.9.5.3.
@startNumber	O	specifies the number of the first Media Segment in this Representation in the Period. For details refer to 5.3.9.5.3.

Segment Base Information		specifies Segment base information.
SegmentTimeline	0...1	specifies the timeline of arbitrary Segment durations For more details see 5.3.9.6.
BitstreamSwitching	0 ... 1	specifies the URL including a possible byte range for the Bitstream Switching Segment. For the type definition refer to Table 13.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

Table 13 — Semantics of elements of type `URLType`

Element or Attribute Name	Use	Description
Element of type `URLType`		defines an HTTP-URL
`@sourceURL`	O	specifies the source URL part and shall be formatted either as an `<absolute-URI>` according to RFC 3986, Clause 4.3, with a fixed scheme of “http” or “https” or as a `<relative-ref>` according to RFC 3986, Clause 4.2. If not present, then any **`BaseURL`** element is mapped to the `@sourceURL` attribute and the range attribute shall be present.
`@range`	O	specifies the byte range restricting the above HTTP-URL. The byte range shall be expressed and formatted as a `byte-range-spec` as defined in RFC 2616, Clause 14.35.1. It is restricted to a single expression identifying a contiguous range of bytes. If not present, the element refers to the entire resource referenced in the `@sourceURL` attribute.

5.3.9.2.3 XML-Syntax

```
<!-- Segment information base -->
<xs:complexType name="SegmentBaseType">
  <xs:sequence>
    <xs:element name="Initialization" type="URLType" minOccurs="0"/>
    <xs:element name="RepresentationIndex" type="URLType" minOccurs="0"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="timescale" type="xs:unsignedInt"/>
  <xs:attribute name="presentationTimeOffset" type="xs:unsignedLong"/>
  <xs:attribute name="timeShiftBufferDepth" type="xs:duration"/>
  <xs:attribute name="indexRange" type="xs:string"/>
  <xs:attribute name="indexRangeExact" type="xs:boolean"  default="false"/>
  <xs:attribute name="availabilityTimeOffset" type="xs:double"/>
  <xs:attribute name="availabilityTimeComplete" type="xs:boolean"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

```
<!-- Multiple Segment information base -->
<xs:complexType name="MultipleSegmentBaseType">
  <xs:complexContent>
    <xs:extension base="SegmentBaseType">
      <xs:sequence>
        <xs:element name="SegmentTimeline" type="SegmentTimelineType" minOccurs="0"/>
        <xs:element name="BitstreamSwitching" type="URLType" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="duration" type="xs:unsignedInt"/>
      <xs:attribute name="startNumber" type="xs:unsignedInt"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- Segment Info item URL/range -->
<xs:complexType name="URLType">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="sourceURL" type="xs:anyURI"/>
  <xs:attribute name="range" type="xs:string"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

5.3.9.3 Segment list

5.3.9.3.1 Overview

The Segment list is defined by one or more **SegmentList** elements. Each **SegmentList** element itself contains a list of **SegmentURL** elements for a consecutive list of Segment URLs. Each Segment URL may contain the Media Segment URL and possibly a byte range. The Segment URL element may also contain an Index Segment.

The semantics of the attributes and elements for the Segment list are provided in 5.3.9.3.2, Table 14. The XML syntax of the Segment Information is provided in 5.3.9.3.3.

5.3.9.3.2 Semantics

Table 14 — Semantics of SegmentList element

Element or Attribute Name	Use	Description
SegmentList		specifies Segment information.
@xlink:href	O	specifies a reference to a remote element entity that contains one or multiple elements of type **SegmentList**
@xlink:actuate	OD default: "onRequest"	specifies the processing set, can be either "onLoad" or "onRequest"
MultipleSegmentBaseInformation		Multiple Segment base information as defined in 5.3.9.2, Table 12.
SegmentURL	0 … N	specifies a Media Segment URL and a possibly present Index Segment URL
@media	O	in combination with the @mediaRange attribute specifies the HTTP-URL for the Media Segment.

Element or Attribute Name	Use	Description
		It shall be formatted as an <absolute-URI> according to RFC 3986, Clause 4.3, with a fixed scheme of "http" or "https" or as a <relative-ref> according to RFC 3986, Clause 4.2. If not present, then any **BaseURL** element is mapped to the @media attribute and the range attribute shall be present.
@mediaRange	O	specifies the byte range within the resource identified by the @media corresponding to the Media Segment. The byte range shall be expressed and formatted as a byte-range-spec as defined in RFC 2616, Clause 14.35.1. It is restricted to a single expression identifying a contiguous range of bytes. If not present, the Media Segment is the entire resource referenced by the @media attribute.
@index	O	in combination with the @indexRange attribute specifies the HTTP-URL for the Index Segment. It shall be formatted as an <absolute-URI> according to RFC 3986, Clause 4.3, with a fixed scheme of "http" or "https" or as a <relative-ref> according to RFC 3986, Clause 4.2. If not present and the @indexRange not present either, then no Index Segment information is provided for this Media Segment. If not present and the @indexRange present, then the @media attribute is mapped to the @index. If the @media attribute is not present either, then any **BaseURL** element is mapped to the @index attribute and the @indexRange attribute shall be present.
@indexRange	O	specifies the byte range within the resource identified by the @index corresponding to the Index Segment. If @index is not present, it specifies the byte range of the Segment Index in Media Segment. The byte range shall be expressed and formatted as a byte-range-spec as defined in RFC 2616, Clause 14.35.1. It is restricted to a single expression identifying a contiguous range of bytes. If not present, the Index Segment is the entire resource referenced by the @index attribute.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded)		

Element or Attribute Name	Use	Description
Note that the conditions only holds without using `@xlink:href`. If linking is used, then all attributes are "optional" and <minOccurs=0> Elements are **`bold`**; attributes are non-bold and preceded with an @.		

5.3.9.3.3 XML-Syntax

```
<!-- Segment List -->
<xs:complexType name="SegmentListType">
  <xs:complexContent>
    <xs:extension base="MultipleSegmentBaseType">
      <xs:sequence>
        <xs:element name="SegmentURL" type="SegmentURLType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute ref="xlink:href"/>
      <xs:attribute ref="xlink:actuate" default="onRequest"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- Segment URL  -->
<xs:complexType name="SegmentURLType">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="media" type="xs:anyURI"/>
  <xs:attribute name="mediaRange" type="xs:string"/>
  <xs:attribute name="index" type="xs:anyURI"/>
  <xs:attribute name="indexRange" type="xs:string"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

5.3.9.4 Segment template

5.3.9.4.1 Overview

The Segment template is defined by the **`SegmentTemplate`** element. In this case, specific identifiers that are substituted by dynamic values assigned to Segments, to create a list of Segments. The substitution rules are provided in 5.3.9.4.4.

The semantics of the attributes and elements for the Segment list are provided in 5.3.9.4.2, Table 15. The XML syntax of the Segment Information is provided in 5.3.9.4.3.

5.3.9.4.2 Semantics

Table 15 — Semantics of `SegmentTemplate` element

Element or Attribute Name	Use	Description
`SegmentTemplate`		specifies Segment template information.
MultipleSegmentBaseInformation		Provides the Multiple Segment base information as defined in 5.3.9.2.
`@media`	O	specifies the template to create the Media Segment List. For more details refer to 5.3.9.4.4.

@index	O	specifies the template to create the Index Segment List. If neither the $Number$ nor the $Time$ identifier is included, this provides the URL to a Representation Index. For more details refer to 5.3.9.4.4.
@initialization	O	specifies the template to create the Initialization Segment. Neither $Number$ nor the $Time$ identifier shall be included. For more details refer to 5.3.9.4.4.
@bitstreamSwitching	O	specifies the template to create the Bitstream Switching Segment. Neither $Number$ nor the $Time$ identifier shall be included. For more details refer to 5.3.9.4.4.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

5.3.9.4.3 XML syntax

```
<!-- Segment Template -->
<xs:complexType name="SegmentTemplateType">
  <xs:complexContent>
    <xs:extension base="MultipleSegmentBaseType">
      <xs:attribute name="media" type="xs:string"/>
      <xs:attribute name="index" type="xs:string"/>
      <xs:attribute name="initialization" type="xs:string" />
      <xs:attribute name="bitstreamSwitching" type="xs:string" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

5.3.9.4.4 Template-based Segment URL construction

The **SegmentTemplate**@media attribute, the **SegmentTemplate**@index attribute, the **SegmentTemplate**@initialization attribute and the **SegmentTemplate**@bitstreamSwitching attribute each contain a string that may contain one or more of the identifiers as listed in Table 16.

In each URL, the identifiers from Table 16 shall be replaced by the substitution parameter defined in Table 16. Identifier matching is case-sensitive. If the URL contains unescaped $ symbols which do not enclose a valid identifier then the result of URL formation is undefined. In this case it is expected that the DASH Client ignores the entire containing **Representation** element and the processing of the MPD continues as if this **Representation** element was not present. The format of the identifier is also specified in Table 16.

Each identifier may be suffixed, within the enclosing '$' characters, with an additional format tag aligned with the printf format tag as defined in IEEE 1003.1-2008 [10] following this prototype:

```
%0[width]d
```

The width parameter is an unsigned integer that provides the minimum number of characters to be printed. If the value to be printed is shorter than this number, the result shall be padded with zeros. The value is not truncated even if the result is larger.

The Media Presentation shall be authored such that the application of the substitution process results in valid Segment URLs.

Strings outside identifiers shall only contain characters that are permitted within URLs according to RFC 3986.

Table 16 — Identifiers for URL templates

$<Identifier>$	**Substitution parameter**	**Format**
$$	Is an escape sequence, i.e. "$$" is replaced with a single "$"	not applicable
$RepresentationID$	This identifier is substituted with the value of the attribute **`Representation`**`@id` of the containing Representation.	The format tag shall not be present.
$Number$	This identifier is substituted with the *number* of the corresponding Segment.	The format tag may be present. If no format tag is present, a default format tag with `width`=1 shall be used.
$*Bandwidth*$	This identifier is substituted with the value of **`Representation`**`@bandwidth` attribute value.	The format tag may be present. If no format tag is present, a default format tag with `width`=1 shall be used.
$*Time*$	This identifier is substituted with the value of the **`SegmentTimeline`**`@t` attribute for the Segment being accessed. Either $Number$ or $Time$ may be used but not both at the same time.	The format tag may be present. If no format tag is present, a default format tag with `width`=1 shall be used.

5.3.9.5 Segment information

5.3.9.5.1 Overview

The *Segment Information* provides the following information:

- the presence or absence of Initialization, Index and Bitstream Switching Segment information
- the HTTP-URL and possibly a byte range for each accessible Segment in each Representation,
- all valid Segment URLs declared by the containing MPD,
- for services with **`MPD`**`@type='dynamic'`, the Segment availability start time and Segment availability end time of each Segment,
- an approximate Media Presentation start time of each Media Segment in the Media Presentation timeline within the Period.

The derivation of Initialization, Media, Index and Bitstream Switching Segment Information from the elements **`SegmentBase`**, **`SegmentList`** and **`SegmentTemplate`** is provided in 5.3.9.5.2, 5.3.9.5.3, 5.3.9.5.4 and

5.3.9.5.5. Reference resolution as defined in 5.6.4 and base URL selection as defined in 5.6.5 using **BaseURL** elements as defined in 5.6 shall be applied to any URLs.

5.3.9.5.2 Initialization Segment information

Each Representation has assigned at most one Initialization Segment.

The presence of an Initialization Segment is indicated by the presence of **SegmentBase.Initialization**, **SegmentList.Initialization**, the **SegmentTemplate.Initialization** element or the **SegmentTemplate**@initialization attribute that may contain URL and byte range information or URL construction rules for the Initialization Segment.

If neither **Initialization** element nor **SegmentTemplate**@initialization attribute are present for a Representation then each Media Segment within the Representation shall be self-initializing.

For services with **MPD**@type='dynamic', the Segment availability start time of the Initialization Segment is the sum of the value of the **MPD**@availabilityStartTime and the *PeriodStart* time as defined in 5.3.2.1 of the containing Period and the Segment availability end time of the Initialization Segment is the largest Segment availability end time of any Media Segment in this Representation. For Segment availability of Media Segments refer to 5.3.9.5.3.

The data structures retrieved from the Initialization URL are defined in 6.2.2.

5.3.9.5.3 Media Segment information

If a Representation consists of more than one Media Segment, then this ach Representation has assigned a list of consecutive Media Segments. The list may be specified explicitly by one or more **SegmentList** elements or implicitly by a **SegmentTemplate** element.

Each entry in this Media Segment list has assigned the following parameters:

- A valid Media Segment URL and possibly a byte range,
- the number and position of the Media Segment in the Representation,
- the MPD start time of the Media Segment in the Representation providing an approximate presentation start time of the Segment in the Period.
- the MPD duration of the Media Segment providing an approximate presentation duration of the Segment.

The MPD start time and the MPD duration may be approximate and do not necessarily reflect the exact Media Presentation time. For more details on the relation of MPD start times and Media Presentation time refer to 7.2.1.

In order to obtain the list of Media Segment URLs, i.e. the URL for each Segment at a specific position *k* in the list based on the Segment Information, the following shall apply:

- if **SegmentTemplate** element is present the Template-based Segment URL construction in 5.3.9.4.4 shall be applied as follows
  - If the Representation contains or inherits a **SegmentTemplate** element with *$Number$* then the URL of the Media Segment at position *k* in the Representation is determined by replacing the $*Number*$ identifier by (*k*-1) + @startNumber.
  - If the Representation contains or inherits a **SegmentTemplate** element with *$Time$* then the URL of the media segment at position *k* is determined by replacing the $*Time*$ identifier by the time address associated to this Segment. The time address is determined as follows:

- if the `@duration` attribute is present, then the time address is determined by replacing the $*Time*$ identifier with ((*k*-1) + (k_{Start}-1))* `@duration` with k_{Start} the value of the `@startNumber` attribute, if present, or 1 otherwise. Further, the media time of the Segment shall be accurately expressed by the MPD information in the following sense:
  - the value ((*k*-1) + (k_{Start}-1))* `@duration` is identical to the earliest presentation time in the segment.
  - the duration of the Segment in media presentation time shall be identical to the value of the `@duration` attribute.
- if the **`SegmentTimeline`** element is present, then the time address shall be determined by replacing the $*Time*$ identifier with the earliest presentation time of the *k*-th segment as documented in the Segment timeline in 5.3.9.6.

- if one or more **`SegmentList`** elements are present they contain itself a list of **`SegmentURL`** elements for a consecutive list of Media Segment URLs. The number of the first Segment in the list within this Period is determined by the value of the **`SegmentList`**`@startNumber` attribute, if present, or it is 1 in case this attribute is not present. The sequence of multiple **`SegmentList`** elements within a Representation shall result in Media Segment List with consecutive numbers.
- none of the above: In this case only a single Media Segment shall be present with the URL provided by a **`BaseURL`** element and the **`SegmentBase`** element may be present.

For the derivation of the MPD start time and duration of each Media Segment in the list of Media Segments, the position *k* of the Media Segment and the following information is used.

- If neither `@duration` attribute nor **`SegmentTimeline`** element is present, then the Representation shall contain exactly one Media Segment. The MPD start time is 0 and the MPD duration is obtained in the same way as for the last Media Segment in the Representation (see below for more details).
- If `@duration` attribute is present, then the MPD start time of the Media Segment is determined as (*k* - (k_{Start} - 1)) times the value of the attribute `@duration` with k_{Start} the value of the `@startNumber` attribute, if present, or 1 otherwise. The MPD duration of the Media Segment is the value of the attribute `@duration` unless the Media Segment is the last one the Representation (see below for more details).
- If `@duration` attribute is not present and the **`SegmentTimeline`** element is present then rules in 5.3.9.6 apply to determine the start time and duration of each Media Segment in the Media Segment list.
- To determine the duration of the only or the last Media Segment of any Representation in a Period, the MPD shall include sufficient information to determine the duration of the containing Period. For example, the **`MPD`**`@mediaPresentationDuration`, or **`Period`**`@duration`, or next **`Period`**`@start` may be present.

For services with **`MPD`**`@type='dynamic'`, the Segment availability start time of a Media Segment is the sum of

- the value of the **`MPD`**`@availabilityStartTime`,
- the *PeriodStart* time of the containing Period as defined in 5.3.2.1,
- the MPD start time of the Media Segment, and
- the MPD duration of the Media Segment.

NOTE: By adding the MPD duration of the segment to the segment availability start time of the segment, the segment availability start time of the first segment of each Period depends on the segment duration. This enables to provide segments in Representations with shorter MPD duration earlier, for example to reduce latency for certain Representations.

The Segment availability end time of a Media Segment is the sum of the Segment availability start time, the MPD duration of the Media Segment and the value of the attribute `@timeShiftBufferDepth` for this Representation.

If the `@availabilityTimeOffset` attribute is present for a Representation, then the adjusted segment availability start time is determined by subtracting the value of `@availabilityTimeOffset` from the Segment availability start time. This adjusted segment availability start time provides a time instant in wall-clock time at which a Segment becomes an available Segment. Note that if the `@availabilityTimeComplete` flag is set to true for such a Representation, then the entire Segment may not yet be available at the adjusted segment availability start time.

The MPD shall include URL information for all Segments with an availability start time less than both (i) the end of the Media Presentation and (ii) the sum of the latest time at which this version of the MPD is available on the server and the value of the **`MPD`**`@minimumUpdatePeriod`.

The data structures retrieved from the URL referring to a Media Segment are defined in 6.2.3.

5.3.9.5.4 Index Segment information

Each Segment typically has assigned Segment Index information that may be provided in an explicitly declared Index Segment.

The presence of explicit Index Segment information is indicated

- by the presence of one **`RepresentationIndex`** element providing the Segment Index for the entire Representation, or
- by the presence of at least one of the two attributes `@index` and `@indexRange` in the **`SegmentList.SegmentURL`** element, or
- by the presence of **`SegmentTemplate`**`@index` attribute. If either $Number$ or $Time$ are present the Template-based Segment URL construction in 5.3.9.4.4 shall be applied with number set to the number of the corresponding Media Segment. If not present, the **`SegmentTemplate`**`@index` attribute constitutes a reference to Representation Index.

The `@indexRange` attribute may also be used to provide the byte range for an index within a Media Segment, where this is allowed by the Media Segment format. In this case the `@index` attribute shall not be present and the range specified shall lie completely within any byte range specified for the Media Segment.

The availability of Index Segments is identical to the availability to the Media Segments they correspond to.

The data structures retrieved from the URL referring to an Index Segment are defined in 6.2.4.

5.3.9.5.5 Bitstream Switching Segment information

Each Representation has assigned at most one Bitstream Switching Segment. The Bitstream Switching Segment is only relevant in case the `@bitstreamSwitching` flag is set to '`true`' and may enable the creation of a conforming Segment sequence for Segments from different Representations.

The presence of a Bitstream Switching Segment is indicated by the presence of the **`BitstreamSwitching`** element or the **`SegmentTemplate`**`@bitstreamSwitching` attribute that that may contain URL and byte range information or construction rules for the URL.

If neither **BitstreamSwitching** element nor **SegmentTemplate**@bitstreamSwitching attribute are present for a Representation and the @bitstreamSwitching flag is set to 'true', there are no Bitstream Switching Segments.

The Segment availability time of the Bitstream Switching Segment is identical to the one specified for the Initialization Segment in 5.3.9.5.2.

The data structures retrieved from the URL referring to a Bitstream Switching Segment are defined in 6.2.5.

5.3.9.6 Segment timeline

5.3.9.6.1 General

The **SegmentTimeline** element expresses the earliest presentation time and presentation duration (in units based on the @timescale attribute) for each Segment in the Representation. The use is an alternative to providing the @duration attribute and provides three additional features:

— the specification of arbitrary Segment durations,

— the specification of accurate Segment durations for one media stream where the duration expresses presentation duration of the Segment, and

— the signalling of discontinuities of the Media Presentation timeline for which no Segment data are present in a specific Representation.

For compactness the syntax of this element includes run-length compression to express a sequence of Segments having constant duration.

The **SegmentTimeline** element shall contain a list of **S** elements each of which describes a sequence of contiguous segments of identical MPD duration. The **S** element contains a mandatory @d attribute specifying the MPD duration, an optional @r repeat count attribute specifying the number of contiguous Segments with identical MPD duration minus one and an optional @t time attribute. The value of the @t attribute minus the value of the @presentationTimeOffset specifies the MPD start time of the first Segment in the series.

The @r attribute has a default value of zero (i.e., a single Segment in the series) when not present. For example, a repeat count of three means there are four contiguous Segments, each with the same MPD duration. The value of the @r attribute of the **S** element may be set to a negative value indicating that the duration indicated in @d is promised to repeat until the **S**@t of the next **S** element or if it is the last **S** element in the **SegmentTimeline** element until the end of the Period or the next update of the MPD, i.e. it is treated in the same way as the @duration attribute for a full period.

Any @d value shall not exceed the value of **MPD**@maxSegmentDuration.

The textual order of the **S** elements within the **SegmentTimeline** element shall match the numbering (and thus time) order of the corresponding Media Segments.

When the **SegmentTemplate** is in use and the $Time$ identifier is present in the **SegmentTemplate**@media then

— at least one Segment Index ('sidx') box shall be present

— the values of the **SegmentTimeline** shall describe accurate timing of each Media Segment. Specifically, these values shall reflect the information provided in the Segment index ('sidx') box, i.e.

  — the value of @timescale shall be identical to the value of the timescale field in the first 'sidx' box,

— the value of **S**@t shall be identical to the value of the earliest_presentation_time in the first 'sidx' box of the Media Segment described in **S**,

— the value of **S**@d shall be identical to sum of the values of all Subsegment_duration fields in the first 'sidx' box of the Media Segment described in **S**.

— The Segment URL for a Media Segment is obtained by replacing the $Time$ identifier by the earliest presentation time obtained from the **SegmentTimeline**.

NOTE As the earliest presentation time of the next Media Segment in the same Representation may be derived from the actual Media Segment, e.g. by the use of the Segment Index, the Segment URL may be generated without reading of the updated MPD that contains the update to the Segment Timeline.

The semantics of the attributes and elements for Segment Timeline are provided in 5.3.9.6.2, Table 17. The XML syntax of the Segment Timeline is provided in 5.3.9.6.3.

5.3.9.6.2 Semantics

Table 17 — Semantics of SegmentTimeline element

Element or Attribute Name	Use	Description
SegmentTimeline		specifies the Segment timeline information
S	1 .. N	specifies Segment start time and duration for a contiguous sequence of segments of identical durations. The textual order of the **S** elements must match the indexed (and thus time) order of the corresponding Media Segments.
@t	O	specifies the MPD start time, in @timescale units, the first Segment in the series starts relative to the beginning of the Period. The value of this attribute must be equal to or greater than the sum of the previous **S** element earliest presentation time and the sum of the contiguous Segment durations. If the value of the attribute is greater than what is expressed by the previous **S** element, it expresses discontinuities in the timeline. If not present then the value shall be assumed to be zero for the first **S** element and for the subsequent **S** elements, the value shall be assumed to be the sum of the previous **S** element's earliest presentation time and contiguous duration (i.e. previous **S**@t + @d * (@r + 1)).
@d	M	specifies the Segment duration, in units of the value of the @timescale.
@r	OD default: 0	specifies the repeat count of the number of following contiguous Segments with the same duration expressed by the value of @d. This value is zero-based (e.g. a value of three means four Segments in the contiguous series). A negative value of the @r attribute of the **S** element indicates that the duration indicated in @d

		attribute repeats until the start of the next **S** element, the end of the Period or until the next MPD update.

Legend:
For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory.
For elements: <minOccurs>...<maxOccurs> (N=unbounded)
Elements are **bold**; attributes are non-bold and preceded with an @.

5.3.9.6.3 XML syntax

```
<!-- Segment Timeline -->
<xs:complexType name="SegmentTimelineType">
  <xs:sequence>
    <xs:element name="S" minOccurs="1" maxOccurs="unbounded" >
      <xs:complexType>
        <xs:attribute name="t" type="xs:unsignedLong"/>
        <xs:attribute name="d" type="xs:unsignedLong" use="required"/>
        <xs:attribute name="r" type="xs:int" use="optional" default="0"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
      </xs:complexType>
    </xs:element>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

5.4 Media Presentation Description updates

If **MPD**@type is set to 'dynamic', the MPD may be updated during the Media Presentation. Updates typically extend the accessible Segment list for each Representation, introduce a new Period, update Segment locations or terminate the Media Presentation.

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.

If the attribute **MPD**@minimumUpdatePeriod is not present, no update to the MPD is expected, the attribute **MPD**@mediaPresentationDuration or the **Period**@duration of the last Period shall be present and the MPD shall remain valid until the Media Presentation end time.

If the attribute **MPD**@minimumUpdatePeriod is present updates to the MPD are expected and restricted in a sense that at the location where the MPD is available at a certain time, the MPD is also valid for the duration of the value of the **MPD**@minimumUpdatePeriod attribute. Specifically the following shall hold.

If the *i*-th version of the MPD is the last version of MPD till the end of the Media Presentation, let *Texp(i)* be the Media Presentation end time. Otherwise, let *Texp(i)* be the sum of the value of

MPD@minimumUpdatePeriod and the wall-clock time at which the *i*-th version of the MPD is updated (and replaced with the (*i*+1)-th version). The *i*-th MPD shall remain valid until *Texp*(*i*) in the following sense:

— all Segments with availability start time less than *Texp*(*i*) shall be available at their availability start times at the location advertised in the *i*-th MPD.

— all Representations have a Segment with an availability start time, *Tavail*, which is less than *Texp*(*i*) and with duration not less than (*Texp*(*i*) - *Tavail*).

NOTE 1 the actual duration of this Segment is not known at the client until this Segment or the updated MPD is fetched and this Segment duration may be less than the previous Segment duration if it is the last Segment in the Period or if the Segment Timeline is in place.

NOTE 2 The clients may not know *Texp*(*i*), but they can each calculate a lower bound on *Texp*(*i*) by adding **MPD**@minimumUpdatePeriod to the wall-clock time at which they request the MPD.

NOTE 3 The second condition above ensures that sufficient media is contained in each Representation to present the Media Presentation up to *Texp*(*i*) for a client that begins playing each Segment at the earliest possible time (its availability start time).

NOTE 4 The result of the MPD validity requirement is that all items a client expects to be able to retrieve (both segments and MPD elements) are guaranteed to be available for retrieval during the periods that the client can expect them to be accessible.

NOTE 5 An MPD may contain no Period element or only an early available Period may be provided. In this case, updates to the MPD are expected in order to provide the start time of the first Period, which coincides with the start of the actual Media Presentation.

NOTE 6 An update of the MPD may not necessarily change the MPD, but may only extend the validity of this MPD.

NOTE 7 If the @minimumUpdatePeriod is set to 0, then as a consequence of NOTE 2, all segments with availability start time less than the request time of the MPD are available at the location advertised in the MPD.

5.5 MPD assembly

5.5.1 Introduction

A mechanism for referencing a *remote element entity* from within a local MPD is defined. A subset of W3C XLINK simple links is defined consisting of

— restricted syntax and semantics as specified in 5.5.2, and

— the processing model as specified in 5.5.3.

If the MPD is updated, then the rules in 5.3 apply to the MPD after dereferencing all remote element entities.

5.5.2 Syntax and semantics

Table 18 provides the XLINK attributes that are used in this part of ISO/IEC 23009 and shall be supported accordingly.

Table 18 — XLINK attributes used in this part of ISO/IEC 23009

Attribute	Comments and Usage
@xlink:type	Identifies the type of W3C XLINK being used. In the context of this part of ISO/IEC 23009, all references shall be W3C XLINK

Attribute	Comments and Usage
	simple links. The attribute `@xlink:type` is optional with fixed setting `@xlink:type="simple"`.
`@xlink:href`	references the remote element entity by an URI as defined in IETF RFC 3986. In the context of this part of ISO/IEC 23009, such URIs shall exclusively be HTTP-URLs. NOTE: For example, cookies as defined in RFC 6265 may be used in order to enable targeted resolution of the same HTTP-URL provided in this attribute. Detailed requirements on the support and usage of such technologies are outside the scope of this standard.
`@xlink:show`	Defines the desired behaviour of the remote element entity once dereferenced from within a MPD as defined in W3C XLINK. In the context of this part of ISO/IEC 23009, the attribute `@xlink:show` is optional with fixed setting `@xlink:show="embed"`. NOTE In W3C XLINK, the behaviour of conforming XLink applications when embedding XML-based ending resources, such as a remote element entity, is not defined. Thus, the actual behaviour for this part of ISO/IEC 23009 is defined in 5.5.3.
`@xlink:actuate`	Defines the desired timing of dereferencing a remote element entity from within a MPD as defined in W3C XLINK. The following attribute values are allowed in this part of ISO/IEC 23009: 3) `onLoad`: an application should dereference the remote element entity immediately on loading the MPD. 4) `onRequest` (default): according to W3C Xlink, an application should dereference the remote element entity only on a post-loading event triggered for the purpose of dereferencing. In the context of this Part of ISO/IEC 23009, the application dereferences the link only for those resources it needs (or anticipates it probably will need) and at the time when it needs the content of the remote element entity for playout. Examples include dereferencing a link in a **`Period`** element when the play-time is expected to enter that Period, dereferencing an Adaptation Set link when it appears to contain Representations that will be needed, and so on.

The restricted schema for XLINK in the context of ISO/IEC 23009 is referred to as "xlink.xsd" in any schema in this part of ISO/IEC 23009 and defined in Annex B.

5.5.3 Processing

The following rules apply to the processing of URI references within `@xlink:href`:

1) URI references to remote element entities that cannot be resolved shall be treated as invalid references and invalidate the URI and all `@xlink` attributes included in the element containing `@xlink:href` shall be removed.

2) URI references to remote element entities that are inappropriate targets for the given reference shall be treated as invalid references (see below for the appropriate targets) and invalidate the URI and all `@xlink` attributes included in the element containing `@xlink:href` shall be removed.

3) URI references to remote element entities that contain another `@xlink:href` attribute with `xlink:actuate` set to `onLoad` are treated as invalid circular references and invalidate the URI and all `@xlink` attributes included in the element containing `@xlink:href` shall be removed.

4) The `xlink:href` may contain a URN as `urn:mpeg:dash:resolve-to-zero:2013`. If this value is present, the element containing the `xlink:href` attribute and all `@xlink` attributes included in the element containing `@xlink:href` shall be removed at the time when the resolution is due.

5) If a URI reference is relative then reference resolution as defined in 5.6.4 shall apply.

The remote element entity referenced with `@xlink:href` within an element of the MPD (referred to as MPD element) shall be formatted according to the following rules:

1) Only a single top-level element type of the same type as the MPD element shall be included in a remote element entity. However, multiple top-level elements of the same type may be included in a remote element entity unless explicitly restricted. If multiple top-level elements are obtained from the remote element entity, the elements shall be in appropriate order and the first element shall replace the MPD element. All additional top-level elements shall be inserted immediately after this element in the order in which they appear.

2) The remote element entity may contain another `@xlink:href` attribute with `@xlink:actuate` set to `onRequest`. In this case the resolution to the referenced remote element entity is expected to happen only when the latter is needed again after processing the content in the returned document.

The remote element entity referenced from within an MPD (referred to as appropriate targets) shall be embedded into the MPD by applying the following rules:

1) If the remote element entity is empty, all `@xlink` attributes shall be removed from the element in the MPD and the remaining attributes and child elements shall not be changed.

2) If the remote element entity is non-empty, the original element in the MPD that contains `@xlink:href` shall be replaced with the content in the remote element entity. If multiple top-level elements are obtained from the remote element entity, the elements shall be in appropriate order and the first element shall replace the MPD element. All other top-level elements shall be inserted immediately after this element in the order in which they are declared.

3) All XLINK attributes initially present in the MPD shall be removed after dereferencing is completed.

4) All resources in the remote element entity referenced by `@xlink:href` shall have an availability end time as specified by **`MPD`**`@availabilityEndTime`.

5.6 Base URL Processing

5.6.1 Overview

The **`BaseURL`** element may be used to specify one or more common locations for Segments and other resources. Reference resolution as defined in 5.6.4 shall be applied to each URL in the MPD. Handling of multiple alternative base URLs is addressed in 5.6.5.

The semantics of the attributes and elements for the Base URL are provided in 5.6.2, Table 19. The XML syntax of the Base URL is provided in 5.6.3.

5.6.2 Semantics

Table 19 — Semantics of `BaseURL` element

Element or Attribute Name	Use	Description
`BaseURL`		A URL that can be used as Base URL. The content of this element is a URI string as described in 5.6.4.
`@serviceLocation`	O	This attribute specifies a relationship between Base URLs such that **`BaseURL`** elements with the same `@serviceLocation` value are likely to have their URLs resolve to services at a common network location, for example a common Content Delivery Network. If not present, no relationship to any other Base URL is known.
`@byteRange`	O	if present specifies HTTP partial GET requests may alternatively be issued by adding the byte range into a regular HTTP-URL based on the value of this attribute and the construction rules in Annex E.2. If not present, HTTP partial GET requests may not be converted into regular GET requests. NOTE Such alternative requests are expected to not be used unless the DASH application requires this. For more details refer to Annex E.
`@availabilityTimeOffset`	O	specifies an offset to define the adjusted segment availability time. For semantics, refer to Table 11. If the value is present in **`SegmentBase`** then this attribute should not be present. If present in **`SegmentBase`** and **`BaseURL`** the value in **`BaseURL`** shall be ignored.
`@availabilityTimeComplete`	O	specifies if all Segments of all associated Representation are complete at the adjusted availability start time. For semantics, refer to Table 11. If the value is present in **`SegmentBase`** then this attribute should not be present. If present in **`SegmentBase`** and **`BaseURL`** the value in **`BaseURL`** shall be ignored.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **`bold`**; attributes are non-bold and preceded with an @.		

5.6.3 XML syntax

```
<!-- Base URL -->
<xs:complexType name="BaseURLType">
  <xs:simpleContent>
    <xs:extension base="xs:anyURI">
      <xs:attribute name="serviceLocation" type="xs:string"/>
      <xs:attribute name="byteRange" type="xs:string"/>
      <xs:attribute name="availabilityTimeOffset" type="xs:double" />
      <xs:attribute name="availabilityTimeComplete" type="xs:boolean" />
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

5.6.4 Reference resolution

URLs at each level of the MPD are resolved according to RFC3986 with respect to the **BaseURL** element specified at that level of the document or the level above in the case of resolving base URLs themselves (the document "base URI" as defined in RFC 3986 Section 5.1 is considered to be the level above the MPD level). If only relative URLs are specified and the document base URI cannot be established according to RFC3986 then the MPD should not be interpreted. URL resolution applies to all URLs found in MPD documents.

In addition to the document level (the level above the MPD level), base URL information may be present on the following levels:

— On MPD level in **MPD.BaseURL** element. For details refer to 5.3.1.2.

— On Period level in **Period.BaseURL** element. For details refer to 5.3.2.2.

— On Adaptation Set level in **AdaptationSet.BaseURL** element. For details refer to 5.3.3.2.

— On Representation level in **Representation.BaseURL**. For details refer to 5.3.5.2.

5.6.5 Alternative base URLs

If alternative base URLs are provided through the **BaseURL** element at any level, identical Segments shall be accessible at multiple locations. In the absence of other criteria, the DASH Client may use the first **BaseURL** element as "base URI". The DASH Client may use base URLs provided in the **BaseURL** element as "base URI" and may implement any suitable algorithm to determine which URLs it uses for requests.

5.7 Program information

5.7.1 Overview

Descriptive information on the program may be provided for a Media Presentation within the **ProgramInformation** element.

When multiple **ProgramInformation** elements are present, the `@lang` attribute shall be present and each element shall describe the Media Presentation sufficiently in the language defined by the value of the `@lang` attribute.

For each language, the program information may specify title, source of the program, copyright information, and a URL to more information.

The semantics of the attributes within the **ProgramInformation** element are provided in Table 20 of 5.7.2. The XML syntax of **ProgramInformation** element is provided in 5.7.3.

5.7.2 Semantics

Table 20 — Program information semantics

Element or Attribute Name	Use	Description
ProgramInformation		specifies descriptive information about the program
`@lang`	O	Declares the language code(s) for this Program Information. The syntax and semantics according to IETF RFC 5646 shall be applied. If not present the value is unknown.

@moreInformationURL	O	If provided, this attribute specifies an absolute URL which provides more information about the Media Presentation. If not present the value is unknown.
Title	0 ... 1	specifies the title for the Media Presentation
Source	0 ... 1	specifies information about the original source (for example content provider) of the Media Presentation.
Copyright	0 ... 1	specifies a copyright statement for the Media Presentation, usually starting with the copyright symbol, unicode U+00A9
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

5.7.3 XML syntax

```
<!-- Program Information -->
<xs:complexType name="ProgramInformationType">
  <xs:sequence>
    <xs:element name="Title" type="xs:string" minOccurs="0"/>
    <xs:element name="Source" type="xs:string" minOccurs="0"/>
    <xs:element name="Copyright" type="xs:string" minOccurs="0"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="lang" type="xs:language"/>
  <xs:attribute name="moreInformationURL" type="xs:anyURI"/>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

5.8 Descriptors

5.8.1 General

The descriptor elements are all structured in the same way, namely they contain a `@schemeIdUri` attribute that provides a URI to identify the scheme and an optional attribute `@value` and an optional attribute `@id`. The semantics of the element are specific to the scheme employed. The URI identifying the scheme may be a URN or a URL.

In this part of ISO/IEC 23009, specific elements for descriptors are defined in 5.8.4.

The MPD does not provide any specific information on how to use these elements. It is up to the application that employs DASH formats to instantiate the description elements with appropriate scheme information. However, this part of ISO/IEC 23009 defines some specific schemes in 5.8.5.

DASH applications that use one of these elements must first define a Scheme Identifier in the form of a URI and must then define the value space for the element when that Scheme Identifier is used. The Scheme Identifier appears in the `@schemeIdUri` attribute.

In the case that a simple set of enumerated values are required, a text string may be defined for each value and this string must be included in the `@value` attribute. If structured data is required then any extension element or attribute may be defined in a separate namespace.

The @id value may be used to refer to a unique descriptor or to a group of descriptors. In the latter case, descriptors with identical values for the attribute @id shall be synonymous, i.e. the processing of one of the descriptors with an identical value for @id is sufficient.

Two elements of type DescriptorType are *equivalent*, if the element name, the value of the @schemeIdUri and the value of the @value attribute are equivalent. If the @schemeIdUri is a URN, then equivalence shall refer to lexical equivalence as defined in clause 5 of RFC 2141. If the @schemeIdUri is a URL, then equivalence shall refer to equality on a character-for-character basis as defined in clause 6.2.1 of RFC3986. If the @value attribute is not present, equivalence is determined by the equivalence for @schemeIdUri only. Attributes and element in extension namespaces are not used for determining equivalence. The @id attribute may be ignored for equivalence determination.

The semantics of the attributes within an element of the type DescriptorType are provided in Table 21 of 5.8.2. The XML schema definition of DescriptorType is provided in 5.8.3. The specific descriptors follow these syntax and semantics.

5.8.2 Semantics of generic descriptor

Table 21 — Semantics of elements of type DescriptorType

Element or Attribute Name	Use	Description
Element of type DescriptorType		specifies a descriptor.
@schemeIdUri	M	specifies a URI to identify the scheme. The semantics of this element are specific to the scheme specified by this attribute. The @schemeIdUri may be a URN or URL. When a URL is used, it should also contain a month-date in the form mmyyyy; the assignment of the URL must have been authorized by the owner of the domain name in that URL on or very close to that date, to avoid problems when domain names change ownership.
@value	O	specifies the value for the descriptor element. The value space and semantics must be defined by the owners of the scheme identified in the @schemeIdUri attribute.
@id	O	specifies an identifier for the descriptor. Descriptors with identical values for this attribute shall be synonymous, i.e. the processing of one of the descriptors with an identical value is sufficient.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

5.8.3 XML syntax of generic descriptor

```
<!-- Descriptor -->
<xs:complexType name="DescriptorType">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="schemeIdUri" type="xs:anyURI" use="required"/>
  <xs:attribute name="value" type="xs:string"/>
  <xs:attribute name="id" type="xs:string"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

5.8.4 Specific descriptors

5.8.4.1 Content protection

For the element **`ContentProtection`** the `@schemeIdUri` attribute is used to identify a content protection scheme. This attribute should provide sufficient information, possibly in conjunction with the `@value` and/or extension attributes and elements, such as the DRM system(s), encryption algorithm(s), and key distribution scheme(s) employed, to enable a client to determine whether it can possibly play the protected content. The **`ContentProtection`** element can be extended in a separate namespace to provide information specific to the content protection scheme (e.g., particular key management systems or encryption methods).

When the **`ContentProtection`** element is not present the content shall not be content-protected.

When multiple **`ContentProtection`** elements are present, each element shall describe a content protection scheme that is sufficient to access and present the Representation.

5.8.4.2 Role

For the element **`Role`** the `@schemeIdUri` attribute is used to identify the role scheme employed to identify the role of the media content component. Roles define and describe characteristics and/or structural functions of media content components.

One Adaptation Set or one media content component may have assigned multiple roles even within the same scheme.

This part of ISO/IEC 23009 defines a simple role scheme in 5.8.5.5.

In addition, this part of ISO/IEC 23009 defines other roles schemes to support signalling for multiple view signals in 5.8.5.6.

5.8.4.3 Accessibility

For the element **`Accessibility`** the `@schemeIdUri` attribute is used to identify the accessibility scheme employed. Accessibility is a general term used to describe the degree to which the DASH Media Presentation is available to as many people as possible.

> NOTE **`Accessibility`** elements fulfill a very similar purpose with respect to media content components as for **`Role`** elements, but are specifically intended for accessibility.

One Adaptation Set or one media content component may have assigned multiple accessibility purposes even within the same scheme.

This part of ISO/IEC 23009 does not define a specific accessibility scheme, but the simple role scheme in 5.8.5.5 may be used to express a minimum amount of accessibility information.

5.8.4.4 Rating

For the element **Rating** the @schemeIdUri attribute is used to identify the rating scheme employed.

Ratings specify that content is suitable for presentation to audiences for which that rating is known to be appropriate, or for unrestricted audiences.

NOTE if an audience has a rating restriction it is intended that content that has associated ratings should not be presented to that audience, unless at least one scheme is recognized and the rating value indicates that the content is appropriate to that audience.

This part of ISO/IEC 23009 does not define a rating scheme.

5.8.4.5 Viewpoint

For the element **Viewpoint** the @schemeIdUri attribute is used to identify the viewpoint scheme employed.

Adaptation Sets containing non-equivalent **Viewpoint** element values contain different media content components. The **Viewpoint** elements may equally be applied to media content types that are not video.

Adaptation Sets with equivalent **Viewpoint** element values are intended to be presented together. This handling should be applied equally for recognized and unrecognized @schemeIdUri values.

This Part of ISO/IEC 23009 does not define a viewpoint scheme.

5.8.4.6 Frame-packing

For the element **FramePacking** the @schemeIdUri attribute is used to identify the frame-packing configuration scheme employed.

Multiple **FramePacking** elements may be present. If so, each element shall contain sufficient information to select or reject the described Representations.

NOTE if the scheme or the value for all **FramePacking** elements are not recognized the DASH client is expected to ignore the described Representations. A client may reject the Adaptation Set on the basis of observing a **FramePacking** element.

The descriptor may carry frame-packing schemes using the URN label and values defined for VideoFramePackingType in ISO/IEC 23001-8.

NOTE: This part of ISO/IEC 23009 also defines frame-packing schemes in 5.8.5.6. These schemes are maintained for backward-compatibility, but it recommended to use the signalling as defined in ISO/IEC 23001-8.

5.8.4.7 Audio channel configuration

For the element **AudioChannelConfiguration** the @schemeIdUri attribute is used to identify the audio channel configuration scheme employed.

Multiple **AudioChannelConfiguration** elements may be present indicating that the Representation supports multiple audio channel configurations. For example, it may describe a Representation that includes MPEG Surround audio supporting stereo and multichannel.

NOTE if the scheme or the value for this descriptor is not recognized the DASH client is expected to ignore the descriptor.

The descriptor may carry audio channel configuration using the URN label and values defined for OutputChannelPosition in ISO/IEC 23001-8.

NOTE: A scheme for audio channel configuration is also defined in 5.8.5.4 of this part of ISO/IEC 23009. This scheme is maintained for backward-compatibility, but it recommended to use the signalling as defined in ISO/IEC 23001-8.

In addition, a scheme for audio channel configuration is defined in 5.8.5.4 of this part of ISO/IEC 23009.

5.8.4.8 Essential Property Descriptor

For the element **`EssentialProperty`** the Media Presentation author expresses that the successful processing of the descriptor is essential to properly use the information in the parent element that contains this descriptor unless the element shares the same `@id` with another **`EssentialProperty`** element.

If **`EssentialProperty`** elements share the same `@id`, then processing one of the **`EssentialProperty`** elements with the same value for `@id` is sufficient. At least one **`EssentialProperty`** element of each distinct `@id` value is expected to be processed.

NOTE if the scheme or the value for this descriptor is not recognized the DASH client is expected to ignore the parent element that contains the descriptor.

Multiple **`EssentialProperty`** elements with the same value for `@id` and with different values for `@id` may be present.

5.8.4.9 Supplemental Property Descriptor

For the element **`SupplementalProperty`** the Media Presentation author expresses that the descriptor contains supplemental information that may be used by the DASH client for optimized processing.

NOTE if the scheme or the value for this descriptor is not recognized the DASH client is expected to ignore the descriptor.

Multiple **`SupplementalProperty`** elements may be present.

5.8.4.10 Asset Identifier

The **`AssetIdentifier`** is used to identify the asset on Period level. If two different Periods contain equivalent Asset Identifiers then the content in the two Periods belong to the same asset.

NOTE if the scheme or the value for this descriptor is not recognized the **`AssetIdentifier`** element may still be used to understand the equivalence of Asset Identifiers across Periods. Processing of the descriptor scheme and value by the DASH client is not essential for normal operation.

5.8.5 Specific scheme definitions

5.8.5.1 General

The definition of specific schemes (both syntax and semantics) to be used in any of the descriptor elements requires the definition of the URI by the authors to link the content description to the Media Presentation. In 5.8.5 some schemes and scheme identifiers are defined to enable usage of existing code points in combination with this part of ISO/IEC 23009 as well as to provide simple means to support different functionalities.

5.8.5.2 Content protection

The following defines a set of URIs that identify specific content protection schemes, i.e. schemes contained in the **`ContentProtection`** element:

— For Representations based on ISO/IEC 14496-12, the following URI is defined to indicate protection schemes identified by a the Scheme Type within the Scheme Type Box of the Protection Scheme Information Box of the file:

`urn:mpeg:dash:mp4protection:2011`

In this scheme, the value of the `@value` attribute shall be the 4CC contained in the Scheme Type Box, suitably escaped according to RFC 2141 and may include the version number. The 4CC and the version number, if present, shall be separated by a ":". The version number shall be encoded as up to 8 hexadecimal digits, where the leading '0's may be omitted.

— For Representations based on ISO/IEC 13818-1 (MPEG-2 Transport Stream), the following URI are is defined to indicate the Conditional Access System used:

`urn:mpeg:dash:13818:1:CA_descriptor:2011`

In this scheme, the value of the `@value` attribute shall be the 4-digit lower-case hexadecimal Representation of the 16-bit CA_system_ID from the `CA_descriptor` as defined in ISO/IEC 13818-1.

— For Representations based on ISO/IEC 14496-12 a content protection scheme using the Protection System Specific Header Box defined in ISO/IEC 23001-7 may be identified in the **`ContentProtection`** element. In this case a UUID URN as defined in RFC 4122 indicating the UUID specified in the SystemId field of the Protection System Specific Header Box shall be used. This does not imply that such schemes cannot define alternative URNs, or that all UUID URNs refer to schemes of this type.

5.8.5.3 Frame-packing

The following defines a set of URIs that identify specific frame-packing arrangements, i.e. schemes contained in the **`FramePacking`** element:

— For Adaptation Sets or Representations that contain a video component that conforms to ISO/IEC 14496-10, the URI `urn:mpeg:dash:14496:10:frame_packing_arrangement_type:2011` is defined. The `@value` shall be value as defined for `VideoFramePackingType` in ISO/IEC 23001-8. .

— For Adaptation Sets or Representations that contain a video component that conforms to ISO/IEC 13818-1, the URI `urn:mpeg:dash:13818:1:stereo_video_format_type:2011` is defined.. The `@value` shall be value as defined for `VideoFramePackingType` in ISO/IEC 23001-8.

5.8.5.4 Audio channel configuration schemes

The following defines a URI that identifies channel configuration signalling for Representations that contain an audio component. The URI "`urn:mpeg:dash:23003:3:audio_channel_configuration:2011`" is defined to indicate the channel configuration as defined by Table 68 (Channel Configurations, meaning of `channelConfigurationIndex`, mapping of channel elements to loudspeaker positions') of ISO/IEC 23003-3. The `@value` shall be the value as defined for `OutputChannelPosition` in ISO/IEC 23001-8.

The URN "`urn:mpeg:dash:outputChannelPositionList:2012`" defines a list of output channel positions to signal individual speaker positions. The `@value` shall be a space-delimited list of values as defined of the `OutputChannelPosition` as defined in ISO/IEC 23001-8. For example, the `@value` for the 7.1 channel configuration 2 high as 2/0/0, 5 mid as 3/0/2 and 0.1 low, where a/b/c indicates speaker count in front, side and back, respectively and 0.1 indicates a subwoofer channel), is "`2 0 1 4 5 3 17 1`".

5.8.5.5 DASH role scheme

The URN "urn:mpeg:dash:role:2011" is defined to identify the role scheme defined in Table 22. Note that **Role**@value shall be assigned to Adaptation Sets that contain a media component type to which this role is associated.

Table 22 — Role@value attribute for scheme with a value "urn:mpeg:dash:role:2011"

Role@value	**Description**
caption	captions (see note 3 below)
subtitle	subtitles (see note 3 below)
main	main media component(s) which is/are intended for presentation if no other information is provided
alternate	media content component(s) that is/are an alternative to (a) main media content component(s) of the same media component type (see note 2 below)
supplementary	media content component that is supplementary to a media content component of a different media component type (see Note 1 below)
commentary	media content component with commentary (e.g. director's commentary) (typically audio)
dub	media content component which is presented in a different language from the original (e.g. dubbed audio, translated captions)

NOTES

1) A normal audio/video program labels both the primary audio and video as "main". However, when the two media component types are not equally important, for example (a) video providing a pleasant visual experience to accompany a music track that is the primary content or (b) ambient audio accompanying a video showing a live scene such as a sports event, that is the primary content, the accompanying media may be assigned a "supplementary" role.

2) alternate media content components should carry other descriptors to indicate in what way it differs from the main media content components (e.g. a Viewpoint descriptor or a Role descriptor), especially when multiple alternate media content components including multiple supplementary media content components are available.

3) open ("burned in") captions or subtitles would be marked as media type component "video" only, but having a descriptor saying "caption" or "subtitle";

5.8.5.6 DASH Multiple views scheme

This scheme is defined for multiple views media content description.

This scheme may be used with the **Role** descriptor applied to a **ContentComponent** element of type video or to an **AdaptationSet** element. If this scheme is applied to an **AdaptationSet** element containing multiple views, each media content component of type video shall have a **ContentComponent** element that contains the **Role** descriptor using this scheme. A **Role** element of this scheme is used to indicate which views amongst the views comprising multiple presentable stereo pairs the contained media content component or components represent. If *N* views are available that can be combined into *M* valid stereo pairs,

the **Role** with @schemeIdURI equal to "urn:mpeg:dash:stereoid:2011" signals which views form a stereo pair and which one is the left view and which one is the right view of each stereo pair. The @value of the Role element shall contain a space-delimited list of view indicators 'l*i*' or 'r*j*' where *i*, *j* are non-negative decimal integers. A stereo pair *i* (0 <= *i* < *M*) is formed by using a view whose **Role** element includes the view indicator 'l*i*' as the left view and a view whose **Role** element contains the view indicator 'r*i*' as the right view. Within the @value attribute view indicators shall be ordered with all left view indictors preceding all right view indicators and within each group in ascending order of view index.

5.9 DASH metrics descriptor

5.9.1 Overview

This part of ISO/IEC 23009 does not define mechanisms for reporting metrics, however it does define a set of metrics and a mechanism that may be used by the service provider to trigger metric collection and reporting at the clients, should a reporting mechanism be available. The trigger mechanism is based on the **Metrics** element in the MPD. The element contains the list of DASH Metrics for which the measurements are desired, the time interval and the granularity for the measurements, as well as the scheme according to which the metric reporting is desired.

The semantics of the attributes within the **Metrics** element are provided in Table 23 of 5.9.2. The XML syntax of **Metrics** element is provided in 5.9.3.

The semantics of the **Reporting** element are provided in 5.9.4.

5.9.2 Semantics

Table 23 — Semantics of Metrics element

Element or Attribute Name	Use	Description
Metrics		DASH metric element
@metrics	M	specifies all DASH Metrics (as a list of DASH Metric keys as defined in Annex D, separated by a comma) that the client is desired to report.
Range	0 ... N	specifies the time period during which DASH Metrics collection is requested. When not present, DASH Metrics reporting is requested for the whole duration of the content.
@starttime	O	specifies the start time of the DASH Metrics collection operation. When not present, DASH Metrics collection is requested from the beginning of content consumption. For services with **MPD**@type='dynamic', the start time is indicated in wall clock time by adding the value of this attribute to the value of the **MPD**@availabilityStartTime attribute. For services with **MPD**@type='static', the start time is indicated in Media Presentation time and is relative to the *PeriodStart* time of the first Period in this MPD. NOTE: For example, if **MPD**@availabilityStartTime is 14:30 and the metrics collection is intended to start at 14:45, then @starttime is 0:15.

Element or Attribute Name	Use	Description
@duration	O	specifies the duration of the DASH metrics collection interval. The value of the attribute expresses in Media Presentation time. If not present, the value is identical to the Media Presentation duration.
Reporting	1 ... N	specifies information about the requested reporting method and formats. For more details refer to 5.9.4.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

5.9.3 XML syntax

```
<!-- Metrics -->
<xs:complexType name="MetricsType">
  <xs:sequence>
    <xs:element name="Reporting" type="DescriptorType" maxOccurs="unbounded"/>
    <xs:element name="Range" type="RangeType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"
  </xs:sequence>
  <xs:attribute name="metrics" type="xs:string" use="required"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- Metrics Range -->
<xs:complexType name="RangeType">
  <xs:attribute name="starttime" type="xs:duration"/>
  <xs:attribute name="duration" type="xs:duration"/>
</xs:complexType>
```

5.9.4 Metric reporting

DASH clients should collect metrics based on the **Metric** element and report the collected metrics using one of the reporting schemes in the **Reporting** descriptor in the **Metrics** element.

It is expected that elements containing unrecognized reporting schemes are ignored by the DASH client.

If multiple **Reporting** elements are present, it is expected that the client processes one of the recognized reporting schemes.

No reporting scheme is specified in this part of ISO/IEC 23009. It is expected that external specifications may define formats and delivery for the reporting data. External specifications defining a reporting scheme should take specific care to respect privacy issues.

5.10 Events

5.10.1 Overview

Events may be provided in the MPD or within a Representation in order to signal aperiodic information to the DASH client or to an application. Events are timed, i.e. each event starts at a specific media presentation time and typically has a duration. Events include DASH specific signalling or application-specific events. In the

latter case, a proper scheme identifier identifies the application such that the DASH client can forward the event to the proper application.

Events of the same type are clustered in Event Streams. This enables a DASH client to subscribe to an Event Stream of interest and ignore Event Streams that are of no relevance or interest.

Two ways of signalling events are provided, namely

— events signalled in the MPD as defined in 5.10.2

— events signalled inband in the Representation as defined in 5.10.3.

DASH-specific events are defined in 5.10.4.

5.10.2 MPD Events

5.10.2.1 Overview

Events may be signalled in the MPD. A sequence of events assigned to the media presentation time may be provided in the MPD on Period level. Events of the same type are summarized in an Event Stream that is specified by an **`EventStream`** element in a **`Period`** element. Events shall terminate at the end of a Period even if the start time is after the Period boundary or duration of the event extends beyond the Period boundary.

The **`EventStream`** element is structured in a similar way as the descriptor defined in 5.8, namely it contains a `@schemeIdUri` attribute that provides a URI to identify the scheme and an optional attribute `@value`. The semantics of the element are specific to the scheme employed. The URI identifying the scheme may be a URN or a URL.

A Period shall contain at most one **`EventStream`** element with the same value of the `@schemeIdUri` attribute and the value of the `@value` attribute, i.e. all Events of one type shall be clustered in one Event Stream.

As Event Streams contain timed events, also a time scale attribute `@timescale` is provided to assign events to a specific media presentation time within the Period. The timed events themselves are described by the **`Event`** element.

This specification does not provide any specific information on how to use Event Streams. It is up to the application that employs DASH formats to instantiate the description elements with appropriate scheme information. However, this part of ISO/IEC 23009 defines some specific schemes in 5.10.4.

> Note: A DASH application that uses one of these elements, defines a Scheme Identifier in the form of a URI and then defines the value space for the element when that Scheme Identifier is used. The Scheme Identifier appears in the `@schemeIdUri` attribute.

The semantics of the attributes within the **`EventStream`** element are provided in Table 24 of 5.10.2.2 and the semantics of the attributes within the **`Event`** element are provided in Table 25 of 5.10.2.2. The XML syntax of **`EventStream`** and **`Event`** element is provided in 5.10.2.3.

5.10.2.2 Semantics

Table 24 — Event Stream Semantics

Element or Attribute Name	Use	Description
`EventStream`		specifies event Stream

@xlink:href	O	specifies a reference to an external **EventStream** element
@xlink:actuate	OD default: onRequest	specifies the processing instructions, which can be either "onLoad" or "onRequest". This attribute shall not be present if the @xlink:href attribute is not present.
@schemeIdUri	M	identifies the message scheme. The string may use URN or URL syntax. When a URL is used, it is recommended to also contain a month-date in the form mmyyyy; the assignment of the URL must have been authorized by the owner of the domain name in that URL on or very close to that date. A URL may resolve to an Internet location, and a location that does resolve may store a specification of the message scheme.
@value	O	specifies the value for the event stream element. The value space and semantics must be defined by the owners of the scheme identified in the @schemeIdUri attribute.
@timescale	O	specifies the timescale in units per seconds to be used for the derivation of different real-time duration values in the **Event** elements. If not present on any level, it shall be set to 1.
Event	0 ... N	specifies one event. For details see Table 25. Events in Event Streams shall be ordered such that their presentation time is non-decreasing.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

Table 25 — Event Semantics

Element or Attribute Name	Use	Description
Event		specifies an event and contains the message of the event, formatted as a string. The content of this element depends on the event scheme.
@presentationTime	OD default: 0	specifies the presentation time of the event relative to the start of the Period. The value of the presentation time in seconds is the division of the value of this attribute and the value of the @timescale attribute. If not present, the value of the presentation time is 0.

@duration	O	specifies the presentation duration of the event. The value of the duration in seconds is the division of the value of this attribute and the value of the @timescale attribute. If not present, the value of the duration is unknown.
@id	O	specifies an identifier for this instance of the event. Events with equivalent content and attribute values in the **Event** element shall have the same value for this attribute.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

5.10.2.3 XML-Syntax

```
<!-- Event Stream -->
<xs:complexType name="EventStreamType">
  <xs:sequence>
    <xs:element name="Event" type="EventType" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute ref="xlink:href"/>
  <xs:attribute ref="xlink:actuate" default="onRequest"/>
  <xs:attribute name="schemeIdUri" type="xs:anyURI" use="required"/>
  <xs:attribute name="value" type="xs:string"/>
  <xs:attribute name="timescale" type="xs:unsignedInt"/>
</xs:complexType>

<!-- Event  -->
<xs:complexType name="EventType">
  <xs:sequence>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
    <xs:attribute name="presentationTime" type="xs:unsignedLong" default="0"/>
    <xs:attribute name="duration" type="xs:unsignedLong"/>
    <xs:attribute name="id" type="xs:unsignedInt"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

5.10.3 Inband Event Signalling

5.10.3.1 Overview

Event streams may be multiplexed with Representations by adding the event messages as part of the Segments. The event streams may be present in selected Representations, in one or several selected Adaptation Set only or in all Representations. For example, one possible configuration is one where only the audio Adaptation Sets may contain inband events.

In order to identify the Representations that carry the event stream, the presence of Events shall be signalled in the MPD as defined in 5.10.3.2.

If more than one Representation carries event streams with same @schemeIdUri and the same @value the streams shall be semantically equivalent, i.e. processing one Representation is sufficient.

The format of the box to signal events in the media stream is provided in 5.10.3.3.

5.10.3.2 MPD Signalling

An inband event stream that is present in a Representation shall be indicated by an **InbandEventStream** element on Adaptation Set or Representation level. The **InbandEventStream** type is defined in 5.10.2, Table 24.

One Representation may contain multiple inband Event streams, each indicated by a separate **InbandEventStream** element.

5.10.3.3 Event message box

5.10.3.3.1 Introduction

The Event Message box ('emsg') provides signalling for generic events related to the media presentation time. The same semantics as for an event defined in the MPD specified in 5.10.2 applies.

The Event Message box ('emsg') also provides signalling that are specific for the DASH operations. The event scheme identifier and the events are defined in 5.10.4.

A Media Segment if encapsulated in ISO BMFF may contain one or more event message (‘emsg’) boxes. If present, any 'emsg' box shall be placed before any ‘moof’ box.

The carriage of event messages in the MPEG-2 TS is described in 5.10.3.3.5.

Event message boxes with scheme identifiers that are not defined in the MPD should not be present. If a DASH client detects an event message box with a scheme that is not defined in MPD, the client is expected to ignore it.

5.10.3.3.2 Definition

Box Type: `emsg’
Container: Segment
Mandatory: No
Quantity: Zero or more

5.10.3.3.3 Syntax

```
aligned(8) class DASHEventMessageBox extends FullBox(‘emsg’, version = 0, flags =
0){
   string            scheme_id_uri;
   string            value;
   unsigned int(32)  timescale;
   unsigned int(32)  presentation_time_delta;
   unsigned int(32)  event_duration;
   unsigned int(32)  id;
   unsigned int(8)   message_data[];
   }
}
```

5.10.3.3.4 Semantics

scheme_id_uri: identifies the message scheme. The semantics and syntax of the message_data[] are defined by the owner of the scheme identified. The string may use URN or URL syntax. When a URL is used, it is recommended to also contain a month-date in the form mmyyyy; the assignment of the URL must have been authorized by the owner of the domain name in that URL on or very close to that date. A URL may resolve to an Internet location, and a location that does resolve may store a specification of the message scheme.

`value`: specifies the value for the event. The value space and semantics must be defined by the owners of the scheme identified in the `scheme_id_uri` field.

`timescale` provides the timescale, in ticks per second, for the time and duration fields within this box;

`presentation_time_delta` provides the Media Presentation time delta of the media presentation time of the event and the earliest presentation time in this segment. If the segment index is present, then the earliest presentation time is determined by the field `earliest_presentation_time` of the first `'sidx'` box. If the segment index is not present, the earliest presentation time is determined as the earliest presentation time of any access unit in the media segment. The timescale is provided in the `timescale` field

`event_duration` provides the duration of event in media presentation time. The timescale is indicated in the `timescale` field. The value `0xFFFF` indicates an unknown duration.

`id`: a field identifying this instance of the message. Messages with equivalent semantics shall have the same value, i.e. processing of any one event message box with the same id is sufficient.

`message_data`: body of the message, which fills the remainder of the message box. This may be empty depending on the above information. The syntax and semantics of this field must be defined by the owner of the scheme identified in the `scheme_id_uri` field.

5.10.3.3.5 Carriage of the Event Message Box in MPEG-2 TS

A Media Segment if encapsulated in MPEG-2 Transport Stream may contain one or more event message (‘`emsg`’) boxes encapsulated into transport stream packets.

Transport stream packets carrying the `emsg` box shall use a reserved fixed PID value of 0x0004.

The transport stream packet carrying the start of the `emsg` box shall have the `payload_unit_start_indicator` field set to `1`, and the packet payload will start with the `emsg` box. The complete `Box.type` field shall be present in this first packet, and the payload size shall be at least 8 bytes.

The continuation of box data will occupy following transport stream packets from the same PID. The last packet carrying the end of the box will be padded using adaptation field stuffing bytes.

A segment shall contain only complete boxes. If `@bitstreamSwitching` is set, and subsegments are used, a subsegment shall contain only complete `emsg` boxes.

For any packet with PID value of 0x0004 the value of the `transport_scrambling_control` field shall be set to '`00`'.

5.10.4 DASH-specific events

5.10.4.1 Overview

DASH specific events that are of relevance for the DASH client are signalled in the MPD. The URN "`urn:mpeg:dash:event:2012`" is defined to identify the event scheme defined in Table 22.

Table 26 — `InbandEventStream`@value attribute for scheme with a value "urn:mpeg:dash:event:2012"

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

5.10.4.2 MPD validity expiration

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

MPD validity expiration shall be signalled for all updates causing an extension of the timeline, except for the following ones:

— The value of the **`MPD`**@minimumUpdatePeriod is changed,

— The value of a **`SegmentTimeline.S`**@r has changed,

— A new **`SegmentTimeline.S`** element is added.

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 shall 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.

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

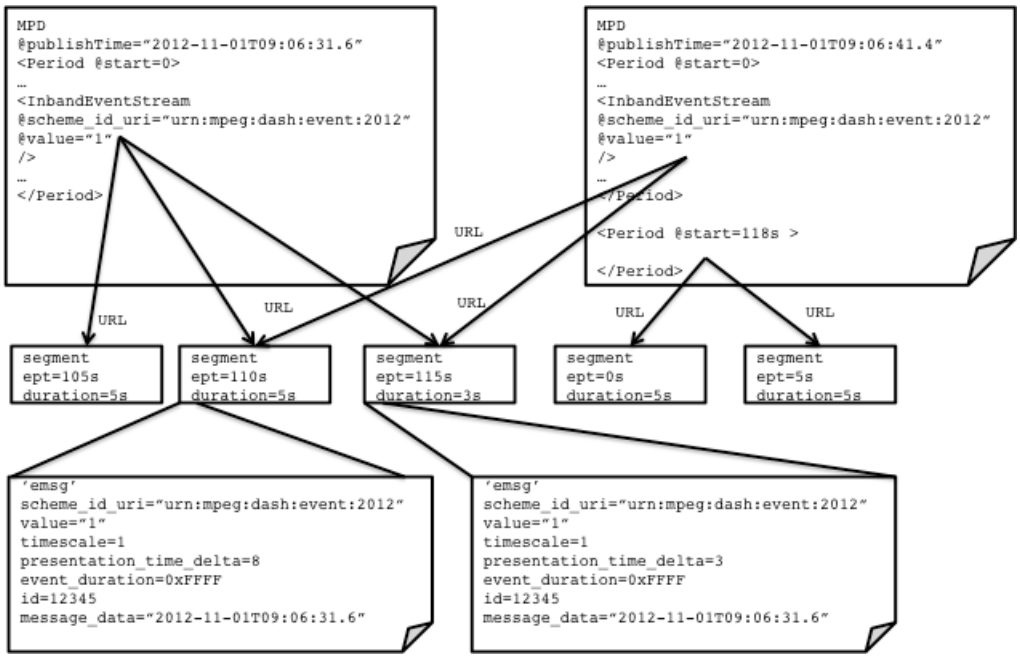

Figure 4 Example for MPD validity expiration to signal new Period

5.10.4.3 MPD Patch

For DASH events with value 2, an MPD patch shall be included in the message. The payload of this message shall be a valid XML patch compliant to XML Patch Operations framework, as defined in IETF RFC 5261.

The result of the patch application shall be parse-tree identical before any xlink resolution to the MPD that would have been retrieved at event time.

XPath selectors shall at least include a check on **`MPD`**`@publishTime`, and the last patch operation should change the value of **`MPD`**`@publishTime`.

NOTE: Additional values for **`InbandEventStream`**`@value` when `@schemeIDURI` is `urn:mpeg:dash:event:2012` are reserved for ISO/IEC.

6 Segment formats

6.1 Introduction

The Segment formats specify the syntax and semantics of the resources that are associated with HTTP-URLs identified by the MPD. For example, an HTTP GET request to a resource identified in the MPD is responded with an HTTP response including an entity body that conforms to a segment format.

Different Segment types are defined in 6.2.

This Part of ISO/IEC 23009 focuses on Segment formats based on MPEG container formats. Specifically,

— in 6.3, Segment formats are described for use with Media Segments based on the ISO Base Media File Format as defined in ISO/IEC 14496-12;

— In 6.4, Segment formats are described for use with Media Segments based on the MPEG-2 Transport Stream as defined in the ISO/IEC 13818-1;

In both cases the Segment formats are defined such that the Media Segment formats comply with the respective container formats.

Guidelines for adding other Segment formats are provided in Annex F.

6.2 Segment types

6.2.1 Introduction

Four different Segment types are defined:

— Initialization Segments containing initialization information for accessing the Representation in 6.2.2,

— Media Segments containing encoded media content components in 6.2.3,

— Index Segments primarily containing indexing information for Media Segments in 6.2.4,

— Bitstream Switching Segments containing essential data to switch to the Representation to which it is assigned in 6.2.5.

6.2.2 Initialization Segment

The Initialization Segment contains initialization information for accessing the Representation. The Initialization Segment shall not contain any media data with an assigned presentation time.

NOTE The Initialization Segment is conceptually processed by the media engine in Figure 2 to initialize the media engines for enabling play-out of Media Segments of the containing Representation.

The Initialization Segment is media format specific and more details shall be defined for each media format that permits or requires the presence of an Initialization Segment.

6.2.3 Media Segment

6.2.3.1 General

A Media Segment contains and encapsulates media streams that are either described within this Media Segment or described by the Initialization Segment of this Representation or both.

In addition, a Media Segment

1) shall contain a number of complete access units.

2) should contain at least one Stream access point (SAP) for each contained media stream.

3) should provide information on how to access the Media Presentation within this Segment, e.g. exact presentation time and an index. There is no requirement that a Media Segment starts with a SAP, but it is possible to signal in the MPD that all media streams in a Segments within a Representation start with a SAP.

4) if it is the first Media Segment in the Representation, it shall contain only media streams that start with a SAP of type 1 or 2.

5) shall contain sufficient information to time-accurately present each contained media component in the Representation without accessing any previous Media Segment in this Representation provided that the Media Segment contains a SAP for each media stream. The time-accuracy enables a client to seamlessly switch Representations and jointly present multiple Representations.

6) may be divided into Subsegments by a Segment Index as defined in 6.2.3.2. In some media formats the Segment Index may be contained in the Media Segment. In other formats the Segment Index may be included in a dedicated Index Segment. For more details on Index Segments refer to 6.2.4.

7) shall specify all Media Presentation times relative to the start of the Period and compensated with the value of the `@presentationTimeOffset`. The presentation time in Media Segments shall be accurate to ensure accurate alignment of all Representations in one Period. For more details refer to 7.2.1.

The Media Segment is media format specific and more details are specified for individual media formats.

6.2.3.2 Subsegments and Segment Index

Media Segments may contain multiple Subsegments. Each Subsegment shall contain a number of complete access units. There may also be media-format-specific restrictions on Subsegment boundaries. If a Segment is divided into multiple Subsegments this division is described by a compact Segment index, which provides the presentation time range in the Representation and corresponding byte range in the Segment occupied by each Subsegment for one or more media streams. Clients may download this index in advance and then issue requests for individual Subsegments.

NOTE Segment Index information is conceptually processed by the DASH access client in Figure 2 in order to access Subsegments by the use of HTTP partial GET requests.

In addition, the Segment Index provides timing and stream access information. This includes the earliest presentation time of access units in each Subsegment of an indexed media stream and the presentation time of the first SAP, if present.

If a Segment Index is present for at least one media stream, then for any media stream for which no Segment Index is present, referred to as non-indexed stream, the following applies:

— every access unit of the non-indexed streams shall be a SAP of type 1.

— for each Subsegment, every non-indexed stream must contain exactly one access unit within the Subsegment with presentation time less than or equal to the earliest presentation time of the Subsegment.

When multiple media streams are indexed in a single index file, the corresponding Segment Index for different media streams should index the same number of Subsegments.

If no Segment Index is provided for a Media Segment, then the Media Segment constitutes one Subsegment.

The Segment Index may be included in the Media Segment, typically in the beginning of the file. Segment Index information may also be provided in separate Index Segments as defined in 6.2.4. A Subsegment may itself be further subdivided using further Segment Indices. If a Subsegment only contains media data but no Segment Index, it is referred to as Media Subsegment.

The Segment Index may contain additional Subsegment indexing information for accessing different levels of Subsegments in a Media Subsegment. For more details refer to 6.2.3.3.

A generic mechanism for indexing of Media Segments is provided by the Segment Index (‘`sidx`’) box in ISO/IEC 14496-12. This indexing applies to all media formats defined in this part of ISO/IEC 23009. In this case,

— the earliest presentation time of a Subsegment is documented in the `earliest_presentation_time` field.

— the byte range is document by the `first_offset` field and the `reference_size` field. If two Segment Index boxes document the same byte range, then the value of their `first_offset` field and their `reference_size` field shall be identical.

6.2.3.3 Subsegment Index

Media Subsegments may be indexed further to enable accessing different levels of Subsegments in a Media Subsegment. This Subsegment Index may also be provided in separate Index Segments together with the Segment Index.

A generic syntax and semantic for Subsegment indexing is provided by the Subsegment Index ('`ssix`') in ISO/IEC 14496-12.

6.2.4 Index Segment

Index Segments contain information that is related to Media Segments and primarily contain indexing information for Media Segments. An Index Segment may provide information for one or more Media Segments.

The Index Segment may be media format specific and more details shall be defined for each media format that permits Index Segments.

6.2.5 Bitstream Switching Segment

A Bitstream Switching Segment contains data essential for switching to the Representation it is assigned to.

The Bitstream Switching Segment is media format specific and more details shall be defined for each media format that permits Bitstream Switching Segments.

6.3 Segment formats for ISO base media file format

6.3.1 Introduction

This Clause defines Segment formats based on the ISO Base Media File Format as specified in ISO/IEC 14496-12. All Segment formats defined in 6.3 shall contain one or more boxes in accordance with the box structure of the ISO base media file format ISO/IEC 14496-12.

Refinements on generic concepts are introduced in 6.3.2. Segment formats are defined for Initialization Segments (6.3.3), Media Segments (6.3.4), and Self-Initializing Media Segments (6.3.5). Bitstream Switching Segments and Index Segments are not defined for this media format.

6.3.2 Preliminaries: Refinements of generic concepts

6.3.2.1 Subsegments

Media Subsegments for Media Segments based on the ISO base media file format are defined as a self-contained set of one or more consecutive movie fragments; such a set contains one or more movie fragment boxes with the corresponding media data ('`mdat`') box(es). A media data box containing data referenced by a movie fragment ('`moof`') box shall follow that movie fragment box and precede the next movie fragment box, if any, containing information about the same track.

For a Media Subsegment, the value of the `reference_type` field in the describing Segment Index ('`sidx`') box shall be set to 0.

6.3.2.2 Media stream access points

Different types of media stream access points for the ISO base media file format are defined in ISO/IEC 14496-12, Annex I.

6.3.2.3 Segment Index

If the Segment Index is provided the Segment Index ('sidx') box in ISO/IEC 14496-12 shall be used. Exact definitions for the use of the Segment Index ('sidx') box with media formats based on the ISO base media file format are specified in ISO/IEC 14496-12.

6.3.2.4 Subsegment Index

If the Subsegment Index is provided the Subsegment Index ('ssix') box in ISO/IEC 14496-12 shall be used. Exact definitions for the use of the Subsegment Index ('ssix') box for the use with media formats based on the ISO base media file format are specified in ISO/IEC 14496-12.

6.3.3 Initialization Segment format

The Initialization Segment shall conform to the ISO base media file format.

The Initialization Segment shall contain an "ftyp" box, and a "moov" box. It shall not contain any "moof" boxes. It may contain other boxes, such as the "pdin" box. The tracks in the "moov" box shall contain no samples (i.e. the entry_count in the "stts", "stsc", and "stco" boxes shall be set to 0), and the "moov" box is thus small.

NOTE This may reduce the start-up time significantly as the Initialization Segment needs to be downloaded before any Media Segment can be processed.

The "mvex" box shall be contained in the "moov" box to indicate that the client has to expect movie fragments. The "mvex" box also sets default values for the tracks and samples of the following movie fragments.

The Initialization Segment provides the client with the metadata that describes the encoding of the media content, specifically of the Representation. The media engine in the client uses the information in the "moov" box to identify the available media content components and their characteristics.

NOTE It is expected that the media engine in the DASH clients does not require any information in the MPD for successful decoding and presentation of the contained media streams.

6.3.4 Media Segment types

6.3.4.1 General

Media Segments can be of different types: simple Media Segments, Indexed Media Segments and Sub-Indexed Media Segments.

All Media Segments shall conform to the general definitions in 6.3.4.2. Additional type-specific constraints are provided further below in 6.3.4.

Further rules on Media Segments in combination with certain MPD attributes are provided in 7.3.

Note that Media Segments may conform to multiple types. Conformance can be expressed by adding the brand(s) to the 'styp' box as a compatible brand and, if applicable, as the major brand.

Unless explicitly mentioned differently, the boxes referred in 6.3.4 are specified in ISO/IEC 14496-12.

6.3.4.2 General format type

A Media Segment conforming to the Media Segment Format for DASH is defined as follows:

— Each Media Segment may contain a 'styp' box and if present shall carry 'msdh' as a compatible brand. The conformance requirement of this brand is defined in this subclause.

— Each Media Segment shall contain one or more whole self-contained movie fragments. A whole, self-contained movie fragment is a movie fragment ('moof') box and a media data ('mdat') box that contains all the media samples that do not use external data references referenced by the track runs in the movie fragment box.

— Each 'moof' box shall contain at least one track fragment.

— The 'moof' boxes shall use movie-fragment relative addressing for media data that does not use external data references, the flag 'default-base-is-moof' shall be set, and data-offset shall be used, i.e. base-data-offset-present shall not be used.

— Absolute byte-offsets shall not be used for this media data. In a movie fragment, the duration by which each track extends should be as close to equal as practical. In particular, as movie fragments are accumulated, the track durations should remain close to each other and there should be no 'drift'.

— Each 'traf' box shall contain a 'tfdt' box.

NOTE 'The track fragment adjustment box 'tfad' as defined in 3GPP TS26.244 may also be present. DASH clients should not apply both the alignment established by the 'tfdt' and the time-shifting implied by the 'tfad', which would result in a double correction.

— Each Media Segment may contain one or more 'sidx' boxes. If present, the first 'sidx' box shall be placed before any 'moof' box and the first Segment Index box shall document the entire Segment.

— For the purpose of determining overlapping and non-overlapping segments, redundant samples as defined in ISO/IEC 14496-12 shall be ignored. In other word, the earliest presentation time of any access unit in the stream shall be computed without taking redundant samples into account.

6.3.4.3 Indexed Media Segment

A Media Segment conforming to the Indexed Media Segment Format is defined as follows:

— Each Media Segment shall comply with the general type as defined in 6.3.4.2 and in addition in each self-contained movie fragment, the movie fragment ('moof') box is immediately followed by its corresponding media data ('mdat').

— Each Media Segment shall contain one or more 'sidx' boxes. The first 'sidx' box shall be placed before any 'moof' box and shall document Subsegments that span the composition time of the entire Segment.

— Each Media Segment shall carry 'msix' as a compatible brand. The conformance requirements of this brand are defined in this subclause.

6.3.4.4 Sub-Indexed Media Segment

A Media Segment conforming to the Sub-Indexed Media Segment Format is defined as follows:

— It shall conform to the indexed Media Segment format as specified in 6.3.4.3.

— The Subsegment Index box ('ssix') shall be present and shall follow immediately after the 'sidx' box that documents the same Subsegment. This immediately preceding 'sidx' shall only index Media Subsegments.

— It shall carry 'sims' in the Segment Type box ('styp') as a compatible brand. The conformance requirements of this brand are defined in this subclause.

6.3.5 Self-Initializing Media Segment formats

6.3.5.1 General format type

The Self-Initializing Media Segment is conformant with the ISO base media file format and defines the DASH Self-Initializing Media Segment 'dsms' brand.

The Self-Initializing Media Segment is conformant with the ISO base media file format.

NOTE Since one Representation only contains one self-initializing Media Segment, switching is expected to happen within the Segment, e.g., at a Subsegment that contains a SAP.

6.3.5.2 Indexed self-initializing Media Segment

The Indexed Self-Initializing Media Segment conforms to the concatenation of an Initialization Segment and a single Indexed Media Segment without the 'styp' box preceding the Media Segment and shall carry 'dash' as a compatible brand.

The format of the Indexed self-initializing Media Segment is a conforming ISO base media file format file and defines the 'dash' brand.

6.4 Segment formats for MPEG-2 transport streams

6.4.1 Introduction

This clause introduces Segment formats that are suitable to be used if Media Segments are valid MPEG-2 TS, conforming to ISO/IEC 13818-1.

NOTE It is possible to encapsulate MPEG-2 TS formatted media within an ISO base media file format. This mode of operation is not discussed in this Subclause. If MPEG-2 TS formatted media is encapsulated in an ISO base media file format, then the rules as defined in 6.3 apply.

Refinements on generic concepts are introduced in 6.4.2. Segment formats are defined for Initialization Segments (see 6.4.3), Media Segments (see 6.4.4), Bitstream Switching Segments (see 6.4.5) and Index Segments (see 6.4.6). MPEG-2 TS specific box structures are defined in 6.4.7.

6.4.2 Preliminaries: Refinements of generic concepts

6.4.2.1 Subsegment

In the context of MPEG-2 TS based delivery formats, a Subsegment is defined as an indexed set of access units consecutive in decode order. A subsegment shall contain complete access units for the indexed media stream (i.e., stream for which reference_ID equals PID), however it may contain incomplete PES packets from other media streams.

These access units are encapsulated in one or more PES packets. Each PES packet is encapsulated into one or more TS packets with the same PID value.

6.4.2.2 Media stream access points

For the case of MPEG-2 TS, a media stream is equivalent to an Elementary Stream as defined in ISO/IEC 13818-1.

Different types of media stream access points are defined in ISO/IEC 14496-12, Annex I. The same type definitions shall apply for the MPEG-2 TS. More specifically, in the case of MPEG-2 TS a SAP corresponds to an Elementary Stream Random Access Point, as defined in ISO/IEC 13818-1. Consequently, I_{SAU} is the position of the first (sync) byte of a TS packet with PID assigned to this Elementary Stream. This TS packet contains the first byte of a PES packet, which, in turn, contains the Elementary Stream Access Point. PES packet starting at I_{SAU} shall contain only an integral number of access units and shall contain a PTS.

NOTE 1 I_{SAU} generally corresponds to the start of a TS packet with PID value for one Elementary Stream, the `payload_unit_start_indicator` field set to `1`, `adaptation_field_control` set to `11`, and the `random_access_indicator` field in the Adaptation Field is set to `1`. For SAP types 1-*3*, the `random_access_indicator` field in the Adaptation Field is commonly set to `1` (this is the case unless no PES payload bytes are found within the packet payload).

NOTE 2 Following the definitions in this Subclause, the first packet of the PCR PID is present at or prior to the TS packet at smallest I_{SAP}. If PCRs are carried on a media PID, the first packet of this PID is the first packet following the initialization data, and carries a PCR. In order to avoid changing the underlying content, the implementer may choose to add a packet carrying only adaptation field with a PCR, but no payload. This packet is placed prior to the smallest I_{SAU} of any stream in this Representation.

NOTE 3 If Index Segment is provided, and the `pcrb` box is present, PCR can be inferred from this box.

6.4.2.3 Segment Index

If the Segment Index is provided the Segment Index (‘`sidx`’) box in ISO/IEC 14496-12 shall be used for Segment Indexing. In addition to these definitions, the following conditions shall be met for a Segment Index used to describe MPEG-2 TS based Media Segment:

— `reference_ID` field of `sidx` box shall be the PID value of the indexed stream.

— All media offsets within `sidx` boxes shall be to the first (sync) byte of a TS packet

NOTE Times within `sidx` boxes are expressed in units of the `timescale` field, rather than in 90KHz clock ticks.

6.4.2.4 Subsegment Index

If the Subsegment Index is provided the Subsegment Index (‘`ssix`’) box in ISO/IEC 14496-12 shall be used for indexing byte ranges within a subsegment. In addition to these definitions, the following conditions shall be met for a Segment Index used to describe MPEG-2 TS based Media Segment.

— All media offsets within `ssix` boxes shall be to the first (sync) byte of a TS packet.

6.4.3 Initialization Segment types and formats

6.4.3.1 Initialization information

Initialization information is any information necessary to enable the media engine to start decoding the payload of any TS packet belonging to any media stream within a (Sub)Segment.

Untimed initialization information includes PAT, CAT, PMT, EMM, and any other PSI information possibly included by the Media Presentation author. Any additional information that does not alter the Media Presentation timeline is allowed.

Time-varying initialization information is information that is required for the successful start of playout, but is different for at least two Subsegments or Segments within a Representation.

Mandatory initialization information summarizes information that shall be present prior to any media data to enable decoding and presentation. As a consequence, mandatory initialization information includes at least the following information, in this order:

— PAT (untimed, unless changes within the Representation);

— PMT (untimed, unless changes within the Representation);

— PCR (time-varying).

If MPEG-2 Conditional Access is used, ECM is considered mandatory untimed initialization information if it does not change for the whole duration of the Period; otherwise it is considered mandatory time-varying initialization information.

6.4.3.2 Initialization Segment

An Initialization Segment shall be a valid MPEG-2 TS, conforming to ISO/IEC 13818-1.

The concatenation of an Initialization Segment with any Media Segment shall have the same presentation duration as the original Media Segment.

The Initialization Segment shall contain mandatory untimed initialization information as defined in 6.4.3.1. Time-varying initialization information shall not be present in the Initialization Segment, i.e.

— PCR-bearing packets shall not be present in the Initialization Segment;

— ECM may be present as long as it does not change within the entire Representation;

— Any PSI table may be present as long as it does not change within the entire Representation.

The Initialization Segment shall contain only complete sections.

Initialization Segment may or may not be present. If it is not present for a given Representation, all Media Segments belonging to this Representation shall be self-initializing. Also, if an Initialization Segment is used, not all initialization information needs to reside in the Initialization Segment, only presence of complete initialization information in the concatenation of Initialization Segment and Media Segment is required.

6.4.4 Media Segment types and formats

6.4.4.1 General

All Media Segments shall conform to the basic Media Segment in 6.4.4.2.

Further rules on Media Segments in combination with certain MPD attributes are provided in 7.4.

6.4.4.2 Basic Media Segment

A Media Segment shall be a valid MPEG-2 TS, conforming to ISO/IEC 13818-1.

As a consequence of the requirement in 5.3.5.1, the concatenation of consecutive Media Segments of the same Representation shall also yield a valid MPEG-2 TS conforming to ISO/IEC 13818-1.

In addition, the following conditions shall be met:

— Media Segments shall contain complete MPEG-2 TS packets,

— Media Segments shall contain exactly one program,

— All time-varying initialization information shall be present between I_{SAP} and I_{SAU} and/or in the Index Segment, if present,

— No Media Segment shall depend on initialization information appearing in any preceding Media Segment.

Media Segments should contain only complete PES packets and sections. Each PES packet should be comprised of one or more complete access units in each packet. Media Segments should contain only complete access units.

6.4.4.3 Content Protection

All information necessary for decrypting, or locating information required to decrypt, the encrypted TS packets in a (Sub)Segment shall be present before the encrypted packet(s) to which they apply, either in the same (Sub)Segment, and/or in the Initialization Segment (if used). As an example, this requires the presence of the ECM necessary for decrypting the first encrypted packet of the (Sub)Segment is within the (Sub)Segment before such a packet. A Subsegment may not have an ECM preceding the first encrypted packet if the location of this ECM can be determined using an Index Segment.

NOTE Sub-Representations may be arranged such that information such as ECM is included in all Sub-Representations that need them, for example by assigning the ECM an individual level and add dependency on all relevant Sub-Representations on this level.

6.4.4.4 Self-initializing Media Segment

A Self-initializing Media Segment conforms to the basic Media Segment as defined in 6.4.4.2 and in addition shall contain at the least all mandatory untimed and timed initialization information as defined in 6.4.3.1.

All required initialization information as defined in 6.4.3.1 should be present prior to any media data.

6.4.5 Bitstream Switching Segment

A Bitstream Switching Segment shall be a valid MPEG-2 TS, conforming to ISO/IEC 13818-1.

A Bitstream Switching Segment when concatenated with any Media Segment shall not alter the Media Presentation timeline for the corresponding Media Segment.

If initialization information is carried within a Bitstream Switching Segment, it shall be identical to the one in the Initialization Segment, if present, of the Representation.

NOTE Authors should use Bitstream Switching Segments when there is a reasonable expectation of non-conforming behaviour (such as continuity counter errors, etc.) at the concatenation point of two consecutive Media Segments from different Representation, lack of correct initialization information (two Representations with different initialization information).

6.4.6 Index Segment

6.4.6.1 General

Index Segments consist of a sequence of ISOBMFF-box-structures.

Index Segments may either be associated to a single Media Segment as specified in 6.4.6.2 or may be associated to all Media Segments in one Representation as specified in 6.4.6.3. An Index Segment may also contain a Subsegment Index as specified in 6.4.6.4 and any other boxes defined in 6.4.7.

It is recommended that Index Segments are at the least provided for one media stream.

NOTE 1 Despite the Media Segments are MPEG-2 TS based, Index Segments are reusing ISOBMFF-box-structures. This allows that the DASH access client in the model of Figure 2 to be universal and independent of the Media Format.

NOTE 2 Index Segments are not valid ISOBMFF files, and complete implementation of ISOBMFF is not necessary to utilize indexes in a MPEG-2 TS based client. A partial implementation would suffice, since only few ISOBMFF boxes, such as 'styp', 'sidx', and 'ssix' are required in order to parse an MPEG-2 TS Index Segment. Other box types may be present in an MPEG-2 TS Index Segment, but if present they shall not contain information required to interpret the `styp`, `sidx` or `ssix` boxes.

6.4.6.2 Single Index Segment

A Single Index Segment indexes exactly one Media Segment and is defined as follows:

- Each Single Index Segment shall begin with a 'styp' box, and the brand 'sisx' shall be present in the 'styp' box. The conformance requirement of the brand 'sisx' is defined in this subclause.
- Each Single Index Segment shall contain one or more 'sidx' boxes which index one Media Segment.
- A Single Index Segment may contain one or multiple 'ssix' boxes. If present, the 'ssix' shall follow the 'sidx' box that documents the same Subsegment without any other 'sidx' preceding the 'ssix'.
- A Single Index Segment may contain one or multiple 'pcrb' boxes as defined in 6.4.7.2. If present, 'pcrb' shall follow the 'sidx' box that documents the same Subsegments, i.e. a 'pcrb' box provides PCR information for every subsegment indexed in the last 'sidx' box.

6.4.6.3 Representation Index Segment

A Representation Index Segment indexes all Media Segments of one Representation and is defined as follows:

- Each Representation Index Segment shall begin with an 'styp' box, and the brand 'risx' shall be present in the 'styp' box. The conformance requirement of the brand 'risx' is defined by this subclause.
- Each Media Segment is indexed by one or more Segment Index box(es); the boxes for a given Media Segment are contiguous;
- Each Segment Index box may be followed by an 'ssix' and/or 'pcrb' box;
- The Segment Index for each Media Segments is concatenated in order, preceded by a single Segment Index box that indexes the Index Segment. This initial Segment Index box shall have one entry in its loop for each Media Segment, and each entry refers to the Segment Index information for a single Media Segment.

The structure of a Representation Index Segment is shown in Figure 5. This figure illustrates a case where a Representation Index Segment is provided and the Subsegment Index is used in order to enable efficient trick mode operation. The figure shows four consecutive Subsegments, S0, S1, S2, and S3, each indexed by an 'sidx' box, and two temporal layers within a video stream, I frames (L0) and P frames (L1), indexed by an 'ssix' box.

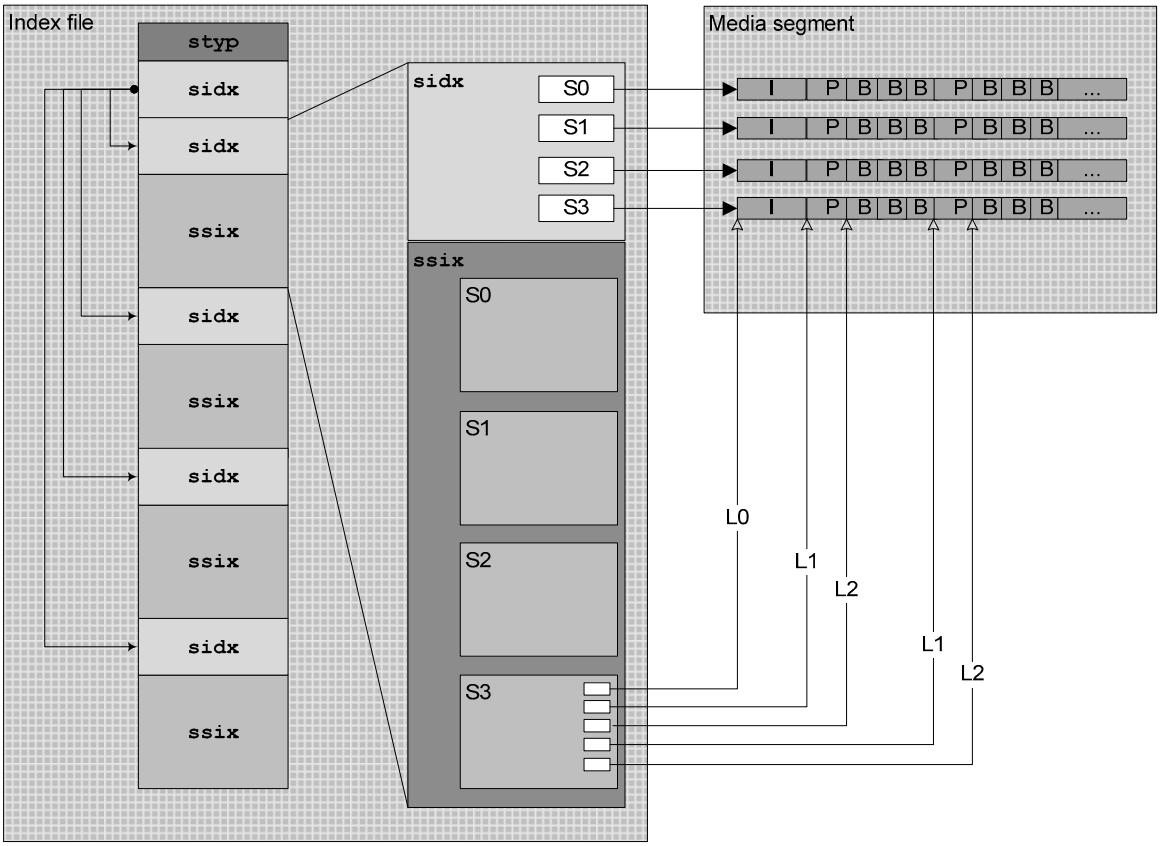

Figure 5 — Structure of Representation Segment Index

6.4.6.4 Subsegment Index Segment

A Subsegment Index Segment shall conform to an Index Segment and also includes a Subsegment Index. A Subsegment Index Segment is defined as follows:

— It shall be either a Single Index Segment or a Representation Index Segment.

— The Subsegment Index box ('`ssix`') shall be present and shall follow immediately after the '`sidx`' box that documents the same Subsegment. The value of the `reference_type` field shall be equal to 0 for this Subsegment in this immediately preceding Segment Index ('`sidx`') box. If the `'pcrb'` box is present, it shall follow `'ssix'`.

— It shall carry '`ssss`' in the Segment Type box ('`styp`') as a compatible brand. The conformance requirement of this brand is defined in this subclause.

6.4.7 Boxes used with MPEG-2 TS Index Segments

6.4.7.1 Introduction

Index Segments may contain additional auxiliary information contained in boxes conforming to the ISO base media file format boxes. Boxes exclusively relevant for the MPEG-2 TS Media Segments are documented in 6.4.7.

6.4.7.2 MPEG-2 TS PCR information box

6.4.7.2.1 Definition

Box Type: 'pcrb'
Container: File
Mandatory: No
Quantity: Zero or one

Signals the PCR information for MPEG-2 TS.

6.4.7.2.2 Syntax

```
aligned(8) class MPEG2TSPCRInfoBox extends Box('pcrb', 0) {
      unsigned int(32)  subsegment_count;
      for( i=1; i <= subsegment_count; i++){
            unsigned int(42)  pcr;
            unsigned int(6)   pad = 0;
      }
}
```

6.4.7.2.3 Semantics

`subsegment_count` is a positive integer specifying the number of Subsegments for which partial Subsegment information is specified in this box. `subsegment_count` shall be equal to `reference_count` in the last Segment Index box.

`pcr` for each iteration of the loop, indicates the MPEG-2 TS PCR corresponding to the first (sync) byte of the first MPEG-2 TS packet in the media Subsegment corresponding to the current iteration. Note, that if this TS packet carries a PCR, its value will be different from the one specified in this field, since ISO/IEC 13818-1 defines PCR as relative to the byte containing the last bit of the `program_clock_reference_base` field.

7 Combined semantics of MPD and Segment formats

7.1 Introduction

An MPD and the referenced Segments comprise a Media Presentation. The formats for these two key components of a DASH-compatible Media Presentation are defined in 5 and 6 of this part of ISO/IEC 23009. In this clause, Media Presentation authoring rules are provided on how the MPD and different Segment formats may be combined to establish a complete Media Presentation.

Specifically aspects are addressed that deal with the Segment, that have special alignment with the Segments of other Representations to enable and simplify seamless switching and joint presentation.

General Media Presentation authoring rules are provided in 7.2 and specific ones for each media format are provided in the remainder of 7. Specifically rules when using the ISO base media file format are provided in 7.3 and the rules when using the MPEG-2 TS are provided in 7.4. Guidelines for other formats are provided in Annex F.

NOTE Representation metadata present in the MPD may also be repeated in the media streams, e.g. in an Initialization Segment or a Media Segment. The Media Presentation shall be provided such that no mismatch between these two values occurs. If it does, the value in the media stream itself takes precedence over values expressed in the MPD, especially when used in the media decoding process.

7.2 General

7.2.1 Media Presentation timeline

One of the key features in DASH is that encoded versions of different media components share a common timeline. The presentation time of access unit within the media content is mapped to the global common presentation timeline for synchronization of different media components and to enable seamless switching of different coded versions of the same media components.

The presentation times within each Period are relative to the *PeriodStart* time of the Period minus the value of the `@presentationTimeOffset`, T_O, of the containing Representation. This means for an access unit with a presentation time T_P signalled in the media stream, the Media Presentation time relative to the *PeriodStart* is $T_M = T_P - T_O$.

Media Segments should not contain any presentation time T_P that is smaller than the value of the `@presentationTimeOffset`, T_O. However, if this is the case, then presentation of the Media Segment is expected to only take place for presentation times greater than or equal to T_O.

The MPD start times as defined in 5.3.9.5.3 shall provide an approximation of the Media Presentation time T_M within the Period. Specifically, the MPD start time shall be drift-free relative to the presentation time T_P signalled in the media stream, i.e. the accuracy of the offset of the MPD start time relative to the presentation time does not depend on the position of the Segment in the Representation.

NOTE At the start of a new Period, the playout procedure of the media content components may need to be adjusted at the end of the preceding Period to match the *PeriodStart* time of the new Period as there may be small overlaps or gaps with a Representation at the end of the preceding Period. Overlaps (respectively gaps) may result from Media Segments with actual presentation duration of the media stream longer (respectively shorter) than indicated by the Period duration. Also in the beginning of a Period if the earliest presentation time T_P of any access unit of a Representation is not equal to T_O then the playout procedures need to be adjusted accordingly.

For the case when **MPD**`@type` is "`dynamic`" and the attribute **MPD**`@suggestedPresentationDelay` is present, then the sum of value of the **MPD**`@availabilityStartTime`, the *PeriodStart* value, the presentation time within the Period of an access unit, T_M, and the value of the attribute **MPD**`@suggestedPresentationDelay` provides a mapping of the presentation time of each access unit to the wall-clock time, for example to express synchronization with a content internal time or for other reasons to enable synchronization of presentation to the wall-clock.

Further media format specific definitions of presentation time may be defined.

7.2.2 Segment Index

If a Segment Index is present in a Media Segment of one Representation within an Adaptation Set, then the following shall hold:

- the order of Segment Index boxes for multiple media streams induces an ordering on the media content components equal to the order in which a Segment Index box for a media stream for each component first appears. This ordering shall be the same for all Segments of all Representations of an Adaptation Set. As a consequence, if there is a Segment Index for a media content component in one Segment there shall be a Segment Index for that media component in all Segments in this Adaptation Set.
- non-indexed media streams in all Representations of an Adaptation Set shall have the same access unit duration.

7.2.3 Segment alignment

No additional requirements beyond those stated in 5.3.3.2 are defined.

7.2.4 Subsegment alignment

No additional requirements beyond those stated in 5.3.3.2 are defined.

7.3 Media Presentation based on the ISO base media file format

7.3.1 General

The Media Presentation as introduced in 5 and 6 is instantiated in this subclause using the ISO base media file format as defined in ISO/IEC 14496-12 as Segment formats.

An ISOBMFF-based DASH Media Presentation is described by an MPD as specified in 5.1. The MIME type of the MPD shall be as defined in Annex C.

The general rules defined in 7.2 shall apply.

The `@mimeType` attribute of each Representation shall be provided according to RFC 4337. Additional parameters may be added according to RFC 6381.

If present, the `@segmentProfiles` shall provide a comma-separated list of the individual Segment profile identifiers.

The following Segment types and formats may be used

- Initialization Segments complying with formats as defined in 6.3.3.
- Media Segments complying with formats as defined in 6.3.4.2.
- Self-Initializing Media Segments complying with formats as defined in 6.3.5.

For ISOBMFF-based Media Presentation the following applies:

1) In all cases for which a Representation contains more than one Media Segment, the following applies:
   i) The Initialization Segment as defined in 6.3.3 shall be present.
   ii) Media Segments shall not be self-initializing. The Media Segment format is defined in 6.3.4.
   iii) If the Media Segment is the last Media Segment in the Representation, this Media Segment may carry the '`lmsg`' compatibility brand. If the Media Segment is not the last Media Segment in the Representation, the '`lmsg`' compatibility brand shall not be present. The '`lmsg`' type is defined in this subclause.
2) In case a Representation contains only a single Media Segment, then one of the following two options are valid.
   - One Initialization Segment as defined in 6.3.3 and one Media Segment as defined in 6.3.4 that is not self-initializing.
   - One Self-Initializing Media Segment as defined in 6.3.5.

Index Segments shall not be present. However, a **`RepresentationIndex`** element or a `@indexRange` attribute may be present to signal the byte range for Segment Index within a Media Segment.

The content authoring rules for the Media Segments in combination with certain MPD attributes for ISOBMFF-based DASH are provided in 7.3.2.

In case Sub-Representations are used, the rules in 7.3.4 shall apply.

7.3.2 Media presentation timeline

The presentation time T_P internal in the media that maps the media to the Media Presentation timeline shall be relative to the movie timeline, i.e. they are composition times after the application of any edit list for the track, as defined in ISO/IEC 14496-12, 8.16.3.

It is recommended that the `@timescale` attribute in the MPD matches the `timescale` field in the Media Header Box of a present track. If the Segment Index ('`sidx`') box is present, then it is further recommended that the track for which the Segment Index ('`sidx`') box that appears first in the Media Segment is the track defining the value of the `@timescale` attribute.

7.3.3 Authoring Rules for specific MPD attributes

7.3.3.1 Segments starting with media stream access points

No additional requirements beyond those stated in 5.3.3.2 and 6.3.2.2 are defined.

7.3.3.2 Bitstream switching

If the `@bitstreamSwitching` is set to '`true`' for a set of Representations within an Adaptation Set, the conditions stated in 5.3.3.2 shall be satisfied and the Bitstream Switching Segment shall not be present.

As a consequence of `@bitstreamSwitching` being set to '`true`', at least the following conditions are satisfied:

- The track IDs for the same media content component are identical for each Representation in each Adaptation Set.
- The conditions required for setting the `@segmentAlignment` attribute to a value other than '`false`' for the Adaptation Set are fulfilled.
- The conditions required for setting (i) the `@startWithSAP` attribute to 2 for the Adaptation Set, or (ii) the conditions required for all Representations within the Adaptation Set to share the same value of `@mediaStreamStructureId` and setting the `@startWithSAP` attribute to 3 for the Adaptation Set, are fulfilled.

7.3.4 Sub-Representations

If a **`SubRepresentation`** element is present in a Representation in the MPD and the attribute **`SubRepresentation`**`@level` is present, then the Media Segments in this Representation shall conform to a Sub-Indexed Media Segment as defined in 6.3.4.4. The Initialization Segment shall contain the Level Assignment ('`leva`') box.

The attribute `@level` specifies the level to which the described Sub-Representation is associated to in the Subsegment Index. The information in Representation, Sub-Representation and in the Level Assignment ('`leva`') box contains information on the assignment of media data to levels.

Media data should be ordered such that each level provides an enhancement compared to the lower levels.

7.4 Media Presentation based on MPEG-2 TS

7.4.1 Introduction

In this subclause, a Media Presentation is instantiated based on Media Segment Formats using the MPEG-2 TS as defined in ISO/IEC 13818-1. A MPEG-2 TS-based DASH Media Presentation is described by an MPD as specified in 5.2. The MIME type of the MPD shall be as defined in Annex C.

The general rules defined in 7.2 shall apply.

The `@mimeType` attribute of each Representation shall be "`video/mp2t`".

The following Segment types and formats may be used

- Initialization Segments complying with formats as defined in 6.4.3.
- Media Segments complying with formats as defined in 6.4.4,
- Bitstream Switching Segments complying with formats as defined in 6.4.5,
- Index Segments complying with formats as defined in 6.4.6.

The `@segmentProfiles` attribute may be absent. If present, it is expected to be ignored.

7.4.2 Media presentation timeline

The presentation time T_P internal in the media that maps the media to the Media Presentation timeline shall be the one defined by the PTS in the MPEG-2 TS.

More specifically, for one Representation, let PTS(i) be the PTS of the i^{th} access unit in the media stream. Furthermore, let $PTS_A(i)$ be PTS(i) adjusted for 33-bit rollovers, i.e. calculated as if PTS had an infinite amount of bits.

T_P calculation is based on differences between PTS(i) and PTS(0), and therefore $T_P(i) = (PTS_A(i) – PTS_0)*S/90000$ with PTS_0 typically PTS(0). With appropriate scaling, PTS_0 be derived from the value of `@presentationTimeOffset` attribute.

NOTE If Index Segment is used, S is provided by in the `timescale` field of the `sidx` box.

If a media stream contains a discontinuity, the $PTS_A(i)$ calculation assumes relative timing is maintained. Therefore, $PTS_A(i)$ will be adjusted by the difference between the value of PCR of the first PCR-bearing packet after the discontinuity and its interpolated PCR value (calculated using the pre-discontinuity PCR rate).

In case of discontinuities it is recommended to add a new Period to reset the value of `@presentationTimeOffset`.

It is recommended that the `@timescale` attribute in the MPD matches the clock frequency S of the elementary streams. If the Segment Index ('`sidx`') box is present, then it is further recommended that the media stream for which the Segment Index ('`sidx`') box that appears first in the Index Segment is the elementary stream defining the value of the `@timescale` attribute.

7.4.3 Authoring rules for specific MPD attributes

7.4.3.1 Segments starting with Media stream access points

No additional requirements beyond those stated in 5.3.3.2 and 6.4.2.2 are defined.

7.4.3.2 Segment alignment

If the `@segmentAlignment` attribute is not set to '`false`', the requirements stated in 5.3.2 and 5.3.3.2 shall be met. In addition, the Media Segment shall contain only complete PES packets and sections and only complete access units for each PID, and the first PES packet shall contain a PTS timestamp.

7.4.3.3 Subsegment alignment

If the `@subsegmentAlignment` flag is not set to '`false`', the semantics as defined in 5.3.3.2 shall apply. In particular, for an MPEG-2 TS-based Media Presentation, a Subsegment shall contain only complete PES packets and sections for each PID, and the first PES packet from each elementary stream shall contain a PTS.

7.4.3.4 Bitstream switching

If `@bitstreamSwitching` flag is set to '`true`' for a set of Representations within an Adaptation Set, then the conditions stated in 5.3.3.2 shall be satisfied. In addition, the conditions in 5.3.3.2 shall not only hold for the entire sequence from i=1,...,M, but for any consecutive sequence of segments with any start index i_S=1,...,M and any end index i_E=i_S,...,M.

If `@bitstreamSwitching` flag is set to '`true`' the Bitstream Switching Segment may be present, indicated by **`BitstreamSwitching`** in the Segment Information. In this case, for any two Representations, X and Y, within the same Adaptation Set, concatenation of Media Segment i of X, Bitstream Switching Segment of Representation Y, and Media Segment $i+1$ of Representation Y shall be a MPEG-2 TS conforming to ISO/IEC 13818-1.

As a consequence of the conformance rule as stated in 5.3.3.2, at least the following conditions are satisfied if `@bitstreamSwitching` flag is set to '`true`':

- The conditions required for setting the `@startWithSAP` attribute to 2 for the Adaptation Set or required for all Representations within the Adaptation Set share the same value of `@mediaStreamStructureId` and setting the `@startWithSAP` attribute of the Adaptation Set 3, are fulfilled.

- The conditions required for setting the `@segmentAlignment` attribute not set to '`false`' for the Adaptation Set are fulfilled.

- PCR shall be present in the Segment prior to the first byte of a TS packet payload containing media data, and not inferred from the `pcrb` box.

7.4.4 Sub-Representations

If a **`SubRepresentation`** element is present in a Representation in the MPD and the **`SubRepresentation`**`@level` is present, then an Index Segment shall be present and shall conform to the format defined in 6.4.6.4.

The Subsegment Index box shall contain at least one entry for the value of **`SubRepresentation`**`@level` and for each value provided in the **`SubRepresentation@`**`dependencyLevel`. The remaining attributes of the **`SubRepresentation`** element should provide sufficient information such that the data contained in the Sub-Representation can be differentiated from the containing Representation as for the MPEG-2 TS no inband assignment of levels is provided.

If Subsegment Index is used for extraction of temporal subsequences, PCR should precede the first bytes of media within the range indicated in the Subsegment index. Also, encryption keys (if used) should not change within the duration of a Subsegment.

8 Profiles

8.1 Definition

Profiles of DASH are defined so as to enable interoperability and the signaling of the use of features.

A profile imposes a set of specific restrictions. Those restrictions are typically on features of the Media Presentation Description (MPD) document and on Segment formats, but may also be on content delivered within Segments, such as on media content types, media format(s), codec(s), and protection formats, or on quantitative measures such as bit-rates, Segment durations and sizes, as well as horizontal and vertical visual presentation size. Profiles defined in this part of ISO/IEC 23009 define restrictions on features of this part of ISO/IEC 23009 and on Segment formats only (e.g. not codec types). Externally defined profiles may additionally impose restrictions on other aspects.

NOTE A profile can also be understood as permission for DASH clients that only implement the features required by the profile to process the Media Presentation (MPD document and Segments). However, as DASH client operation is not specified normatively in this part of ISO/IEC 23009, it is also unspecified how a DASH client conforms to a particular profile. Hence, profiles merely specify restrictions on MPD and Segments rather than DASH client behaviour.

A profile has an identifier, which is a URI. The profiles with which an MPD complies are indicated in the **MPD**@profiles attribute as a comma-separated list of profile identifiers. Profile identifiers defined in this Part of ISO/IEC 23009 are URNs and shall conform to RFC3406. Externally defined profiles may use profile identifiers that are URNs or URLs. When a URL is used, it should also contain a month-date in the form mmyyyy; the assignment of the URL must have been authorized by the owner of the domain name in that URL on or very close to that date, to avoid problems when domain names change ownership.

An MPD is conforming when it satisfies the following:

1. The MPD is valid in terms the schema defined in Annex B.
2. The MPD conforms to the normative requirements defined in this part of ISO/IEC 23009.
3. The MPD conforms to each of the profiles indicated in the **MPD**@profiles attribute as specified below.

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

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

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

1. *ProfA* is included in the **MPD**@profiles attribute.
2. The profile-specific MPD for *ProfA* is valid in terms of the schema defined in Annex B.
3. The profile-specific MPD for *ProfA* conforms to the normative semantics defined in this part of ISO/IEC 23009.
4. The profile-specific MPD for *ProfA* conforms to the restrictions specified for *ProfA*.

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

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

NOTE In other words, each MPD contains at least one Representation in each Period, which fulfills the requirements of a profile listed in **MPD**@profiles. There may be stricter rules on the occurrence of Representations in the specified profiles. For example, it can be required that there is at least one Representation for each media type that contains or is inferred to have the profile identifier of a specific profile.

This part of ISO/IEC 23009 defines six profiles.

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

Three profiles are defined relying on the ISO base media FF as Segment formats. Both, the ISO Base media file format On Demand profile defined in 8.3 and the ISO Base media file format live profile defined in 8.4 are a subset of the ISO Base media file format main profile defined in 8.5. Two profiles are defined for MPEG-2 TS based Media Segment formats: The MPEG-2 TS simple profile defined in 8.7 is a subset of the MPEG-2 TS main profile defined in 8.6. All profiles are a subset of the full profile is defined in 8.2.

8.2 Full profile

8.2.1 General

The full profile includes all features and Segment Types defined in this part of ISO/IEC 23009.

The full profile is identified by the URN "urn:mpeg:dash:profile:full:2011".

8.2.2 Media Presentation Description constraints

The Media Presentation Description shall conform to the following constraints:

— The rules for the MPD as defined in 7.3 or 7.4 shall apply.

— The elements and attributes listed in 5.2.3.2 may be ignored.

8.2.3 Segment format constraints

Representations and Segment formats shall conform to the following constraints:

— Representations shall comply either with the formats defined in 7.3, referring to the Segment formats in 6.3, or to the formats defined in 7.4, referring to the Segment formats in 6.4

8.3 ISO Base media file format On Demand profile

8.3.1 General

This profile is intended to provide basic support for On-Demand content. The primary constraints imposed by this profile are the requirement that each Representation is provided as a single Segment, that Subsegments are aligned across Representations within an Adaptation Set and that Subsegments must begin with Stream Access Points. This permits scalable and efficient use of HTTP servers and simplifies seamless switching.

The On-Demand profile is identified by the URN "urn:mpeg:dash:profile:isoff-on-demand:2011".

8.3.2 Media Presentation Description constraints

The Media Presentation Description shall conform to the following constraints:

- The rules for the MPD and the segments as defined in 7.3 shall apply.
- Representations not inferred to have `@profiles` equal to the profile identifier as defined in 8.3.1 may be ignored.

NOTE A necessary condition to comply with the restrictions defined in 7.3 is that the `@mimeType` equals `video/mp4`, `audio/mp4`, or `application/mp4`.

- The elements and attributes listed in 5.2.3.2 may be ignored.
- **`MPD`**`@type` shall be "`static`".
- The **`Subset`** element may be ignored.
- neither the **`Period.SegmentList`** element nor the **`Period.SegmentTemplate`** element shall be present
- for Adaptation Sets conforming to this profile
  - if either the **`AdaptationSet.SegmentList`** or the **`AdaptationSet.SegmentTemplate`** element is present in an **`AdaptationSet`** element then this **`AdaptationSet`** element may be ignored.
  - if either the **`Representation.SegmentList`** or the **`Representation.SegmentTemplate`** element is present in a **`Representation`** element then this **`Representation`** element may be ignored.
  - if the **`Representation`** element does not contain a **`BaseURL`** element then this **`Representation`** element may be ignored.
  - **`AdaptationSet`** elements with **`AdaptationSet`**`@subsegmentAlignment` not present, or set to '`false`' may be ignored.
  - **`Representation`** elements with a `@subsegmentStartsWithSAP` value absent, zero or greater than 3 may be ignored.
  - **`Representation`** elements with `@subsegmentStartsWithSAP` value equal to 3 may be ignored if both the following conditions hold:
    - the containing Adaptation Set contains more than one Representation, and
    - no other Representation has the same value for `@mediaStreamStructureId`.
- Elements using the `@xlink:href` attribute may be ignored from the MPD. The Representations conforming to this profile are those not accessed through an Adaptation Set that uses an `@xlink:href`.

8.3.3 Segment format constraints

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

- Representations shall comply with the formats defined in 7.3, referring to the Segment formats in 6.3.

— Each Representation shall have one Segment that complies with the Self-Initializing Media Segment as defined in 6.3.5.2.

— All Segment Index ('sidx') and Subsegment Index ('ssix') boxes shall be placed before any Movie Fragment ('moof') boxes.

8.4 ISO Base media file format live profile

8.4.1 General

This profile is optimized for live encoding and may achieve latency of a few seconds by encoding and immediate delivery of short Segments consisting of one or more movie fragments of ISO file format, typically with relatively short duration. Each movie fragment may be requested as soon as available using a template generated URL, so it is not normally necessary to request an MPD update prior to each Segment request. Segments are constrained so that accessing Representations at Segment boundaries is enabled and seamless switching within one Adaptation Set may be performed by first processing (i.e. downloading, decoding and presenting) the come-from Representations and then processing the go-to Representation. . Note that despite the profile is optimized for live services, the MPD@Type attribute may be set to 'static' to distribute non-live content, for example in case a live Media Presentation is terminated, but kept available as On-Demand service.

The ISO Live profile is identified by the URN "urn:mpeg:dash:profile:isoff-live:2011".

8.4.2 Media Presentation Description constraints

The Media Presentation Description shall conform to the following constraints:

— The rules for the MPD and segments as defined in 7.3 shall apply.

— The elements and attributes listed in 5.2.3.2 may be ignored.

— Representations not inferred to have @profiles equal to the profile identifier as defined in 8.4.1 may be ignored.

— In addition, **Representation** elements contained in an **AdaptationSet** element complying to this profile shall have the following constraints:

  — **Representation** elements with @startWithSAP value (either supplied directly or inherited from the containing **AdaptationSet**) equal to 3 may be ignored if both the following conditions hold:

    — the containing Adaptation Set contains more than one Representation, and

    — no other Representation has the same value for @mediaStreamStructureId.

  — The **SegmentTemplate** element shall be present on at least one of the three levels, the Period level containing the Representation, the Adaptation Set containing the Representation, or on Representation level itself.

  — **Representation** elements with a @startWithSAP value (either supplied directly or inherited from the containing) absent, zero or greater than 3 may be ignored.

  — **AdaptationSet** elements with a @segmentAlignment value 'false' or absent may be ignored.

- **Representation** elements with @startWithSAP value (either supplied directly or inherited from the containing Adaptation Set) equal to 3 may be ignored if both of the following conditions hold:
  - the containing Adaptation Set contains more than one Representation, and
  - no other Representation has the same value for @mediaStreamStructureId.
- **Subset** elements may be ignored.
- Elements using the @xlink:href attribute may be ignored from the MPD. The Representations conforming to this profile are those not accessed through an Adaptation Set that uses an @xlink:href.
- When the MPD is updated, the value of **MPD**@availabilityStartTime shall be the same in the original and the updated MPD

8.4.3 Segment format constraints

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

- Representations shall comply with the formats defined in 7.3, referring to the Segment formats in 6.3.
- Each Representation shall have one Initialization Segment and at least one Media Segment.
- Media Segments containing multiple Media Components shall comply with the formats defined in 6.3.4.3, i.e. the brand 'msix'.
- In Media Segments, all Segment Index ('sidx') and Subsegment Index ('ssix') boxes shall be placed before any Movie Fragment ('moof') boxes.

8.5 ISO Base media file format main profile

8.5.1 General

This profile is identified by the URN "urn:mpeg:dash:profile:isoff-main:2011".

8.5.2 Media Presentation Description constraints

The Media Presentation Description shall conform to the following constraints:

- The rules for the MPD and segments as defined in 7.3 shall apply.
- The elements and attributes listed in 5.2.3.2 may be ignored.
- Representations not inferred to have @profiles equal to the profile identifier as defined in 8.5.1 may be ignored.
- The **Subset** element may be ignored.
- Elements using the @xlink:href attribute may be ignored from the MPD. The Representations conforming to this profile are those not accessed through an Adaptation Set that uses an @xlink:href.
- for Adaptation Sets conforming to this profile

— **Representation** elements with a @startWithSAP value greater than 3 or contained in an **AdaptationSet** element with @subsegmentStartsWithSAP value greater than 3 may be ignored.

— If **MPD**@type is 'dynamic',

  — **AdaptationSet** elements with a @segmentAlignment value 'false' or absent may be ignored;

  — **Representation** elements with a @startWithSAP value (either supplied directly or inherited from the containing **AdaptationSet**) absent or zero may be ignored;

  — **Representation** elements with @startWithSAP value (either supplied directly or inherited from the containing **AdaptationSet**) equal to 3 may be ignored if both the following conditions hold:

    — the containing Adaptation Set contains more than one Representation, and

    — no other Representation has the same value for @mediaStreamStructureId.

8.5.3 Segment format constraints

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

— Representations shall comply with the formats defined in 7.3, referring to the Segment formats in 6.3.

— At least one SAP of type 1 to 3, inclusive, shall be present for each track in each Subsegment.

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

— Each Media Segment of the Representations not having @startWithSAP present or having @startWithSAP value 0 or greater than 3 shall comply with the formats defined in 6.3.4.3, i.e. the brand 'msix'. .

8.6 MPEG-2 TS main profile

8.6.1 General

This profile imposes little constraint on the Media Segment format for MPEG-2 Transport Stream content.

This profile is identified by the URN "urn:mpeg:dash:profile:mp2t-main:2011".

8.6.2 Media Presentation Description constraints

The Media Presentation Description shall conform to the following constraints:

— The rules for the MPD as defined in 7.4 shall apply.

— The elements and attributes listed in 5.2.3.2 may be ignored.

— Representations not complying with the restrictions defined in 7.4 or not inferred to have @profiles equal to the profile identifier as defined in 8.6.1 may be ignored.

— Representations not in group 0 may be ignored;

— **Subset** may be ignored;

— Representations containing the **SegmentTimeline** element may be ignored;

— It shall be possible to present a presentation conforming to this profile without resolving @xlink:href in **AdaptationSet** or **SegmentList** elements. Any initial **Period** elements using @xlink:href may be ignored, and the first non-excluded Period must have an explicit @start attribute. After the first non-excluded Period, there shall be no Period using @xlink:href.

— When the MPD is updated, the value of **MPD**@availabilityStartTime shall be the same in the original and the updated MPD.

8.6.3 Segment format constraints

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

— Representations shall comply with the formats defined in 7.4, referring to the Segment formats in 6.4.

8.6.4 Comments and recommendations

The following may be used, if desired:

— Representations not complying with the restrictions defined in 7.4 may still be present, but the presentation should be presentable if they are ignored;

— Both, **SegmentTemplate** or **SegmentList** elements may be used; the normal case would be the use of **SegmentList** elements, however clients should be capable of handling **SegmentTemplate** elements.

For Representations conforming to this profile, it is recommended that:

— Index Segments are supplied.

— **AdaptationSet** elements containing Representations conforming to this profile should not set the value of the @segmentAlignment attribute (either supplied directly or inherited from the containing **MPD**) to 'false'.

— Representations conforming to this profile should set the value of the @startWithSAP to 1 or 2. @startWithSAP may be set to 3 if @mediaStreamStructureId is identical across Representations.

8.7 MPEG-2 TS simple profile

8.7.1 General

This profile is a subset of MPEG-2 TS main profile as defined in 8.6. It poses more restrictions on content encoding and multiplexing in order to allow simple implementation of seamless switching. This is achieved by guaranteeing that a media engine conforming to ISO/IEC 13818-1 can play any bitstream generated by concatenation of consecutive segments from any Representation within the same Adaptation Set.

This profile is identified by the URN "urn:mpeg:dash:profile:mp2t-simple:2011".

8.7.2 Media Presentation Description constraints

The Media Presentation Description shall conform to the following constraints:

— All MPD constraints of MPEG-2 TS Main Profile as defined in 8.6.2 shall be obeyed;

— The elements and attributes listed in 5.2.3.2 may be ignored.

— Representations not complying with the restrictions defined in 7.4 or not inferred to have `@profiles` equal to the profile identifier as defined in 8.7.1 may be ignored.

— If an Index Segment is provided, any Adaptation Set with `@subsegmentAlignment` set to '`false`' may be ignored;

— Any Adaptation Set, which contains more than one Representation and has `@bitstreamSwitching` not set to '`true`', may be ignored;

— When the MPD is updated, the value of **`MPD`**`@availabilityStartTime` shall be the same in the original and the updated MPD.

8.7.3 Segment format constraints

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

— Representations shall comply with the formats defined in 7.4, referring to the Segment formats in 6.4.

— All Media Segment constraints of MPEG-2 TS main profile as defined in 8.6.3 shall be obeyed.

— PSI information, including versions, shall be identical within all Representations contained in an AdaptationSet;

— If MPEG-2 Conditional Access framework is used, same ECM shall be valid for the whole Subsegment, or for the whole Segment if Index Segment is not present.

— For an Index Segment, any single Segment Index ('`sidx`') box may either reference media, or other '`sidx`', but the same '`sidx`' box may not reference both.

8.7.4 Recommendations

For Representations conforming to this profile, it is recommended that:

— Index Segments are supplied,

— **`SegmentTemplate`** elements are used.

Annex A
(informative)

Example DASH client behaviour

A.1 Introduction

The information on client behaviour is purely informative and does not imply any normative procedures on DASH client implementations. However, this information may serve as a guideline to better understand certain features of the formats in the normative parts of this part of ISO/IEC 23009.

A.2 Overview

A DASH client is guided by the information provided in the MPD. This example assumes that the **MPD**@type is 'dynamic'. The behaviour in case **MPD**@type being 'static' is basically a subset of the description here.

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

The following example client behaviour 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. The selection of Adaptation Sets may also take into account information provided by the **AdaptationSet**@group attribute and any constraints of a possibly present **Subset** element.

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. Then the client creates a list of accessible Segments for each Representation for the actual client-local time *NOW* measured in wall-clock time taking into account the procedures introduced in A.3.

3) The client accesses the content by requesting entire Segments or byte ranges of Segments. The client requests Media Segments of the selected Representation by using the generated Segment list.

4) The client buffers media of for at least value of @minBufferTime attribute duration before starting the presentation. Then, once it has identified a Stream Access Point (SAP) for each of the media streams in the different Representations, it starts rendering (in wall-clock-time) of this SAP not before **MPD**@availabilityStartTime + *PeriodStart* + T_{SAP} and not after **MPD**@availabilityStartTime + *PeriodStart* +T_{SAP} + @timeShiftBufferDepth provided the observed throughput remains at or above the sum of the @bandwidth attributes of the selected Representations (if not, longer buffering may be needed). For services with **MPD**@type='dynamic', rendering the SAP at the sum of **MPD**@availabilityStartTime + *PeriodStart* +T_{SAP} and the value of **MPD**@suggestedPresentationDelay is recommended, especially if synchronized play-out with other devices adhering to the same rule is desired.

5) Once the presentation has started, the client continues consuming the media content by continuously requesting Media Segments or parts of Media Segments. The client may switch Representations taking into account updated MPD information and/or updated information from its environment, e.g. change of observed throughput. With any request for a Media Segment containing a stream access point, the client may switch to a different Representation. Seamless switching can be achieved, as the different Representations are time-aligned. Advantageous switching points are announced in the MPD and/or in the Segment Index, if provided.

6) With the wall-clock time *NOW* advancing, the client consumes the available Segments. As *NOW* advances the client possibly expands the list of available Segments for each Representation according to the procedures specified in A.3 If the following conditions are both true, an updated MPD should be fetched:

   i) if the attribute **MPD**`@minimumUpdatePeriod` is present and

   ii) the current playback time gets within a threshold (typically described by at least the sum of the value of the `@minBufferTime` attribute and the value of the `@duration` attribute (or the equivalent value in case the **`SegmentTimeline`** is used) of the media described in the MPD for any consuming or to be consumed Representation.

7) If the clauses in 6) are true, the client should fetch a new MPD, and update *FetchTime*. Once received the client takes into account the possibly updated MPD and the new *FetchTime* in the regeneration of the accessible Segment list for each Representation.

In the following clauses a brief overview on Segment list generation, seeking, support for trick modes and switching Representations are provided.

A.3 Segment list generation

A.3.1 General

Assume that the DASH client has access to an MPD. This clause describes how a client may generate a Segment list for one Representation as shown in Table A.1 from an MPD obtained at *FetchTime* at a specific client-local time *NOW*. In this description, the term *NOW* is used to refer to "the current value of the clock at the reference client when performing the construction of an MPD Instance from an MPD". A client that is not synchronized with a DASH server, which is in turn is expected to be synchronized to UTC, may experience issues in accessing Segments as the Segment availability times provided by the server and the local time *NOW* may not be synchronized. Therefore, DASH clients are expected to synchronize their clocks to a globally accurate time standard.

Table A.1 — Segment list in example client

Parameter Name	Cardinality	Description
Segments	1	Provides the Segment URL list.
InitializationSegment	0, 1	Describes the Initialization Segment. If not present each Media Segment is self-initializing.
URL	1	The URL where to access the Initialization Segment (the client may add a byte range to the URL request if one is provided in the MPD).
MediaSegment	1 … N	Describes the accessible Media Segments.
startTime	1	The MPD start time of the Media Segment in the Period relative to the start time of Period.
duration	1	The MPD duration for the Segment
URL	1	The URL where to access the Media Segment, possibly combined with a byte range.
IndexSegment	1 … N	Describes the accessible Index Segments, if present.
URL	1	The URL where to access the Index Segment, possibly combined with a byte range.

According to 5.3.9 there exist three different ways to describe and generate a Segment List. This description focuses on the first two where either a **SegmentList** element or a **SegmentTemplate** element is present. The case with a single Media Segment using **BaseURL** element and **SegmentBase** element is considered straightforward.

Segments are available at its assigned URL if at wall-clock time *NOW* the Segment availability start time is smaller than or equal to *NOW* and the Segment availability end time is larger than or equal to *NOW*.

Furthermore, assume that for a Representation in a Period, the Segment list is indexed with i=1, ..., N.

A.3.2 Period Start and End Times

Assume that for an MPD with fetch time *FetchTime*

- the *MediaPresentationDuration* is provided either as the value of **MPD**@mediaPresentationDuration if present, or as the sum of *PeriodStart* + **Period**@duration of the last Period.
- the Period start time is provided as *PeriodStart* according to 5.3.2.1 for any Period in the MPD.
- the Period end time referred as *PeriodEnd* is determined as follows: For any Period in the MPD except for the last one, the *PeriodEnd* is obtained as the value of the *PeriodStart* of the next Period. For the last Period in the MPD:
  - if the **MPD**@minimumUpdatePeriod attribute is not present, then *PeriodEnd* is defined as the end time of the Media Presentation, i.e. **MPD**@availabilityStartTime + *MediaPresentationDuration*.
  - if the **MPD**@minimumUpdatePeriod attribute is present, then *PeriodEnd* is defined as the smaller value of *FetchTime* + **MPD**@minimumUpdatePeriod and **MPD**@availabilityStartTime + *MediaPresentationDuration*.

A.3.3 Start Time and Duration

In case the Segment base information contains the @duration attribute, then

- the regular duration d is obtained as d = @duration/@timescale,
- the MPD start time MediaSegment[i].startTime is obtained as (i-1)*d,
- the MPD duration MediaSegment[i].duration is obtained as d unless this Segment is the last Segment in this Period, then the MediaSegment[i].duration is obtained as *PeriodEnd* - MediaSegment.StartTime[i]).

In case the Segment base information contains a **SegmentTimeline** element with N_s **S** elements referred as s=1, ..., N_s, then

- the t[s] is the value of @t of the s-th **S** element divided by the value of the @timescale attribute,
- the o is the value of @presentationTimeOffset for this Representation divided by the value of the @timescale attribute,
- the d[s] is the value of @d of the s-th **S** element divided by the value of the @timescale attribute,
- if the value of @r is greater equal than 0

- the *r*[*s*] is one more than the value of `@r` of the *s*-th **S** element and
- *N*=0
- for *s=1, ... N_s*
  - *N* = *N* + 1
  - MediaSegment[*N*].startTime = *t*[*s*] - *o*
  - MediaSegment[*N*].duration = *d*[*s*]
  - for *j* = 1, ..., *r*[*s*]
    - *N* = *N* + 1
    - MediaSegment[*N*].startTime = MediaSegment[*N-1*].startTime + *d*[*s*]
    - MediaSegment[*N*].duration = *d*[*s*]
- else
  - the MPD duration MediaSegment[*i*].duration is obtained as *d*[0] unless this Segment is the last Segment in this Period, then the MediaSegment[*i*].duration is obtained as *PeriodEnd* - MediaSegment.StartTime[*i*]).

If neither the `@duration` nor the **`SegmentTimeline`** element is given, then

- *N*=1,
- MediaSegment.startTime[*1*] = 0,
- MediaSegment.duration[*1*] = *PeriodEnd* - *PeriodStart*,

If the Representation contains or inherits one or more **`SegmentList`** elements, providing a set of explicit URL(s) for Media Segments, then all *N* Segment URLs are provided.

If the Representation contains or inherits a **`SegmentTemplate`** element with *$Number$* then the URL of the Media Segment *i*, MediaSegment.URL[*i*], is obtained by replacing the $*Number*$ identifier by *i* + `@startNumber` in the **`SegmentTemplate`**`@media` string.

If the Representation contains or inherits a **`SegmentTemplate`** element with *$Time$* then the URL of the Media Segment *i*, MediaSegment.URL[*i*], is obtained by replacing the $*Time*$ identifier by MediaSegment[*i*].startTime in the **`SegmentTemplate`**`@media` string.

A.3.4 Media Segment list restrictions

The Media Segment List is restricted to a list of accessible Media Segments, which may be a subset of the Media Segments of the complete Media Presentation. The construction is governed by the current value of the clock at the client *NOW* which is greater than or equal to the *FetchTime* of the MPD.

Segments may only be accessed during their Segment availability times. Generally, Any Segment may only be available for any time *NOW* between **`MPD`**`@availabilityStartTime` and **`MPD`**`@availabilityEndTime`. For times *NOW* outside this window, no Segments are available.

In addition, for services with **`MPD`**`@type='dynamic'`, the Segment availability start time T_{avail}[*i*] for a Segment *i* in a specific Period is determined as **`MPD`**`@availabilityStartTime` + *PeriodStart* +

MediaSegment[*i*].startTime + MediaSegment[*i*].duration and the Segment availability end time is determined as **MPD**@availabilityStartTime + *PeriodStart* + MediaSegment[*i*].startTime + @timeshiftBufferDepth + 2*MediaSegment[*i*].duration.

In case of MPD updates, assume the variable *CheckTime* associated to an MPD with *FetchTime* is defined as the sum of the fetch time of this operating MPD and the value of the attribute **MPD**@minimumUpdatePeriod , i.e. *CheckTime* = *FetchTime* + **MPD**@minimumUpdatePeriod. The *CheckTime* is defined on the MPD-documented media time axis; when the client's playback time reaches *CheckTime* - **MPD**@minBufferTime it should fetch a new MPD.

Therefore, based on an MPD that was fetched at fetch time *FetchTime* and has associated a check time *CheckTime*, the largest index i_{max} that is accessible at time *NOW* for the last Period in the MPD is $i_{max} = \max_i \{ T_{avail}[i] <= \min(\text{CheckTime}, NOW) \}$.

A.4 Seeking

Assume that a client attempts to seek to a specific Media Presentation time T_M in a Representation relative to the *PeriodStart* time. According to 7.2.1, the presentation times within each Period are relative to the *PeriodStart* time of the Period minus the value of the @presentationTimeOffset, T_O, of the containing Representation.

Based on the MPD, the client has access to the MPD start time and Media Segment URL of each Segment in the Representation, along with Index Segment URL, if present. The Segment number of the Segment most likely to contain media samples for Media Presentation time T_M is obtained as the maximum Segment index *i**, for which the start time MediaSegment[*i*].startTime is smaller or equal to the T_M. The Segment URL is obtained as MediaSegment[*i**].URL.

Note that timing information in the MPD may be approximate due to issues related to placement of Stream Access Points, alignment of media tracks and media timing drift. As a result, the Segment identified by the procedure above may begin at a time slightly after T_M and the media data for presentation time may be in the previous Media Segment. In case of seeking, either the seek time may be updated to equal the first sample time of the retrieved Media Segment, or the preceding Media Segment may be retrieved instead. However, note that during continuous playout, including cases where there is a switch between alternative versions, the media data for the time between T_M and the start of the retrieved Segment is always available.

For accurate seeking to a presentation time T_M, the DASH Client needs to access Stream Access Points (SAP). To determine the SAP in a Media Segment in case of DASH, the client may, for example, use the information in the Segment Index if present to locate the stream access points and the corresponding presentation time in the Media Presentation.

In the case that the Media Presentation is based on the ISO base media file format and a Segment is a movie fragment, it is also possible for the client to use information within the 'moof' and 'mdat' boxes, for example, to locate Stream Access Points in the media and obtain the necessary presentation time from the information in the movie fragment and the Segment start time derived from the MPD. If no SAP with presentation time before the requested presentation time T_M is available, the client may either access the previous Segment or may just use the first SAP as the seek result. When Media Segments start with a SAP, these procedures are simplified.

In the case that the Media Presentation is based on MPEG-2 TS, the presentation units corresponding to the desired presentation time T_M can be identified by using the indexing information, if present, in conjunction with the differential value of the presentation time stamps (PTS) within the Media Segment. For example, if $T_{M,S}$ denotes the presentation time corresponding to the last SAP leading the desired seek time *tp*, with a corresponding PTS denoted as PTS^s, then the desired seek position within the media has a PTS expressed as: $((T_M - T_{M,S})*\text{timescale}+PTS_S\%2^{33}$.

Also note that not necessarily all information of the Media Segment needs to be downloaded to access the presentation time T_M. The client may for example initially request the Segment Index from the beginning of the Media Segment using partial HTTP GET. By use of the Segment Index, Segment timing can be mapped to

byte ranges of the Segment. By continuously using partial HTTP GET requests, only the relevant parts of the Media Segment may be accessed for improved user experience and low start-up delays.

A.5 Support for trick modes

The client may pause or stop a Media Presentation. In this case client simply stops requesting Media Segments or parts thereof. To resume, the client sends requests to Media Segments, starting with the next Subsegment after the last requested Subsegment.

If a specific **Representation** or **SubRepresentation** element includes the @maxPlayoutRate attribute, then the corresponding Representation or Sub-Representation may be used for the fast-forward trick mode. The client may play the Representation or Sub-Representation with any speed up to the regular speed times the specified @maxPlayoutRate attribute with the same decoder profile and level requirements as the normal playout rate. If a specific **Representation** or **SubRepresentation** element includes the @codingDependency attribute with value set to 'false', then the corresponding Representation or Sub-Representation may be used for both fast-forward and fast-rewind trick modes.

Sub-Representations in combination with Index Segments and Subsegment Index boxes may be used for efficient trick mode implementation. Given a Sub-Representation with the desired @maxPlayoutRate, ranges corresponding to **SubRepresentation**@level all level values from **SubRepresentation**@dependencyLevel may be extracted via byte ranges constructed from the information in Subsegment Index Box. These ranges can be used to construct more compact HTTP GET request.

A.6 Switching Representations

Based on updated information during an ongoing Media Presentation, a client may decide to switch Representations. Switching to a “new” Representation is equivalent to tuning in or seeking to the new Representation from the time point where the "old" Representation has been presented. Once switching is desired, the client should seek to a SAP for each media stream in the “new” Representation at a desired presentation time *tp* later than and close to the current presentation time. Presenting the “old” Representation up to the SAP in the “new” Representation enables seamless switching.

If @segmentAlignment is set true and the @startWithSAP is set to 1, 2 or 3 (and in the latter case the **Representation**@mediaStreamStructureId is identical for the two Representations), then the client may switch at any Segment boundary by just concatenating Segments with consecutive indices from different Representations. No overlap downloading and decoding is required.

The same can be achieved on Subsegment level with @subsegmentAlignment set true and @subsegmentStartsWithSAP the same values and conditions as above.

A.7 Reaction to error codes

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 2616) to the error code. In particular, clients should handle redirections (such as 301 and 307) as these may be used as part of normal operation.

If the DASH access client receives a repeated HTTP error for the request of an MPD, the appropriate response may involve terminating the streaming service.

If the DASH access client receives an HTTP client error (i.e. messages with 4xx error code) for the request of an Initialization Segment, the Period containing the Initialization Segment may not be available anymore or may not be available yet.

Similarly, if the DASH access client receives an HTTP client error (i.e. messages with 4xx error code) for the request of a Media Segment, the requested Media Segment may not be available anymore or may not be available yet. In both these case the client should check if the precision of the time synchronization to a globally accurate time standard 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.

A.8 Encoder clock drift control

Non-alignment between the end of a Representation in one Period and the start time of the next Period may be caused by encoder clock inaccuracy. The client should align the Media Presentation time at each Period start. In addition, significant deviations of the start time of Segments to the media time should be detected and drift-compensating measures may be applied even before the start of the next Period is reached.

Over a longer operation time, a difference in clock accuracy of the encoder and decoder may cause the playback to lag behind real-time or to interrupt temporarily due to the client trying to access data faster than real-time.

For ISO base media file format based, clients may avoid these anomalies by using the Producer Reference Time boxes as follows. The pace r_1 of the encoder clock in relation to the UTC is recovered from Producer Reference Time boxes. If the relative pace r_1 is less than 1, equal to 1, or greater than 1, the encoder clock runs more slowly than the UTC, at an identical pace compared to the UTC, or faster than the UTC, respectively. The pace r_2 of the receiver playout clock in relation to UTC is created by accessing a UTC source. A timescale multiplication factor c is equal to r_1/r_2. A presentation time on a timeline of the receiver playout clock is derived for each sample or access unit by multiplying the composition time of the sample (as indicated by the file format structures) or the presentation time of the access unit (as indicated by the respective Program Elementary Stream header) by the timescale multiplication factor c.

In case of MPEG-2 TS segments, PCR-based drift control may be used.

Annex B
(normative)

MPD schema

The schema of the MPD for this edition of the standard is provided below.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="urn:mpeg:dash:schema:mpd:2011" attributeFormDefault="unqualified"
elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="urn:mpeg:dash:schema:mpd:2011">

  <xs:import namespace="http://www.w3.org/1999/xlink" schemaLocation="xlink.xsd"/>

  <xs:annotation>
    <xs:appinfo>Media Presentation Description</xs:appinfo>
    <xs:documentation xml:lang="en">
      This Schema defines the Media Presentation Description for MPEG-DASH.
    </xs:documentation>
  </xs:annotation>

  <!-- MPD: main element -->
  <xs:element name="MPD" type="MPDtype"/>

  <!-- MPD Type -->
  <xs:complexType name="MPDtype">
    <xs:sequence>
      <xs:element name="ProgramInformation" type="ProgramInformationType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="BaseURL" type="BaseURLType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Location" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Period" type="PeriodType" maxOccurs="unbounded"/>
      <xs:element name="Metrics" type="MetricsType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="xs:string"/>
    <xs:attribute name="profiles" type="xs:string" use="required"/>
    <xs:attribute name="type" type="PresentationType" default="static"/>
    <xs:attribute name="availabilityStartTime" type="xs:dateTime"/>
    <xs:attribute name="availabilityEndTime" type="xs:dateTime"/>
    <xs:attribute name="publishTime" type="xs:dateTime"/>
    <xs:attribute name="mediaPresentationDuration" type="xs:duration"/>
    <xs:attribute name="minimumUpdatePeriod" type="xs:duration"/>
    <xs:attribute name="minBufferTime" type="xs:duration" use="required"/>
    <xs:attribute name="timeShiftBufferDepth" type="xs:duration"/>
    <xs:attribute name="suggestedPresentationDelay" type="xs:duration"/>
    <xs:attribute name="maxSegmentDuration" type="xs:duration"/>
    <xs:attribute name="maxSubsegmentDuration" type="xs:duration"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Presentation Type enumeration -->
  <xs:simpleType name="PresentationType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="static"/>
      <xs:enumeration value="dynamic"/>
    </xs:restriction>
  </xs:simpleType>

  <!-- Period -->
  <xs:complexType name="PeriodType">
    <xs:sequence>
      <xs:element name="BaseURL" type="BaseURLType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="SegmentBase" type="SegmentBaseType" minOccurs="0"/>
      <xs:element name="SegmentList" type="SegmentListType" minOccurs="0"/>
      <xs:element name="SegmentTemplate" type="SegmentTemplateType" minOccurs="0"/>
      <xs:element name="AssetIdentifier" type="DescriptorType" minOccurs="0"/>
      <xs:element name="EventStream" type="EventStreamType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="AdaptationSet" type="AdaptationSetType" minOccurs="0"
```

```
maxOccurs="unbounded"/>
      <xs:element name="Subset" type="SubsetType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="xlink:href"/>
    <xs:attribute ref="xlink:actuate" default="onRequest"/>
    <xs:attribute name="id" type="xs:string"/>
    <xs:attribute name="start" type="xs:duration"/>
    <xs:attribute name="duration" type="xs:duration"/>
    <xs:attribute name="bitstreamSwitching" type="xs:boolean" default="false"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Event Stream -->
  <xs:complexType name="EventStreamType">
    <xs:sequence>
      <xs:element name="Event" type="EventType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="xlink:href"/>
    <xs:attribute ref="xlink:actuate" default="onRequest"/>
    <xs:attribute name="schemeIdUri" type="xs:anyURI" use="required"/>
    <xs:attribute name="value" type="xs:string"/>
    <xs:attribute name="timescale" type="xs:unsignedInt"/>
  </xs:complexType>


  <!-- Event  -->
  <xs:complexType name="EventType">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="presentationTime" type="xs:unsignedLong" default="0"/>
        <xs:attribute name="duration" type="xs:unsignedLong"/>
        <xs:attribute name="id" type="xs:unsignedInt"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <!-- Adaptation Set -->
  <xs:complexType name="AdaptationSetType">
    <xs:complexContent>
      <xs:extension base="RepresentationBaseType">
        <xs:sequence>
          <xs:element name="Accessibility" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
          <xs:element name="Role" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="Rating" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="Viewpoint" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="ContentComponent" type="ContentComponentType" minOccurs="0"
maxOccurs="unbounded"/>
          <xs:element name="BaseURL" type="BaseURLType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="SegmentBase" type="SegmentBaseType" minOccurs="0"/>
          <xs:element name="SegmentList" type="SegmentListType" minOccurs="0"/>
          <xs:element name="SegmentTemplate" type="SegmentTemplateType" minOccurs="0"/>
          <xs:element name="Representation" type="RepresentationType" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute ref="xlink:href"/>
        <xs:attribute ref="xlink:actuate" default="onRequest"/>
        <xs:attribute name="id" type="xs:unsignedInt"/>
        <xs:attribute name="group" type="xs:unsignedInt"/>
        <xs:attribute name="lang" type="xs:language"/>
        <xs:attribute name="contentType" type="xs:string"/>
        <xs:attribute name="par" type="RatioType"/>
        <xs:attribute name="minBandwidth" type="xs:unsignedInt"/>
        <xs:attribute name="maxBandwidth" type="xs:unsignedInt"/>
        <xs:attribute name="minWidth" type="xs:unsignedInt"/>
        <xs:attribute name="maxWidth" type="xs:unsignedInt"/>
        <xs:attribute name="minHeight" type="xs:unsignedInt"/>
        <xs:attribute name="maxHeight" type="xs:unsignedInt"/>
        <xs:attribute name="minFrameRate" type="FrameRateType"/>
        <xs:attribute name="maxFrameRate" type="FrameRateType"/>
        <xs:attribute name="segmentAlignment" type="ConditionalUintType" default="false"/>
        <xs:attribute name="subsegmentAlignment" type="ConditionalUintType" default="false"/>
        <xs:attribute name="subsegmentStartsWithSAP" type="SAPType" default="0"/>
```

```
        <xs:attribute name="bitstreamSwitching" type="xs:boolean"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <!-- Ratio Type for sar and par -->
  <xs:simpleType name="RatioType">
    <xs:restriction base="xs:string">
      <xs:pattern value="[0-9]*:[0-9]*"/>
    </xs:restriction>
  </xs:simpleType>

  <!-- Type for Frame Rate -->
  <xs:simpleType name="FrameRateType">
    <xs:restriction base="xs:string">
      <xs:pattern value="[0-9]*[0-9](/[0-9]*[0-9])?"/>
    </xs:restriction>
  </xs:simpleType>


  <!-- Conditional Unsigned Integer (unsignedInt or boolean) -->
  <xs:simpleType name="ConditionalUintType">
    <xs:union memberTypes="xs:unsignedInt xs:boolean"/>
  </xs:simpleType>

  <!-- Content Component -->
  <xs:complexType name="ContentComponentType">
    <xs:sequence>
      <xs:element name="Accessibility" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Role" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Rating" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="Viewpoint" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="xs:unsignedInt"/>
    <xs:attribute name="lang" type="xs:language"/>
    <xs:attribute name="contentType" type="xs:string"/>
    <xs:attribute name="par" type="RatioType"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Representation -->
  <xs:complexType name="RepresentationType">
    <xs:complexContent>
      <xs:extension base="RepresentationBaseType">
        <xs:sequence>
          <xs:element name="BaseURL" type="BaseURLType" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element name="SubRepresentation" type="SubRepresentationType" minOccurs="0"
maxOccurs="unbounded"/>
          <xs:element name="SegmentBase" type="SegmentBaseType" minOccurs="0"/>
          <xs:element name="SegmentList" type="SegmentListType" minOccurs="0"/>
          <xs:element name="SegmentTemplate" type="SegmentTemplateType" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="id" type="StringNoWhitespaceType" use="required"/>
        <xs:attribute name="bandwidth" type="xs:unsignedInt" use="required"/>
        <xs:attribute name="qualityRanking" type="xs:unsignedInt"/>
        <xs:attribute name="dependencyId" type="StringVectorType"/>
        <xs:attribute name="mediaStreamStructureId" type="StringVectorType"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <!-- String without white spaces -->
  <xs:simpleType name="StringNoWhitespaceType">
    <xs:restriction base="xs:string">
      <xs:pattern value="[^\r\n\t \p{Z}]*"/>
    </xs:restriction>
  </xs:simpleType>


  <!-- SubRepresentation -->
  <xs:complexType name="SubRepresentationType">
    <xs:complexContent>
      <xs:extension base="RepresentationBaseType">
        <xs:attribute name="level" type="xs:unsignedInt"/>
        <xs:attribute name="dependencyLevel" type="UIntVectorType"/>
```

```
        <xs:attribute name="bandwidth" type="xs:unsignedInt"/>
        <xs:attribute name="contentComponent" type="StringVectorType"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <!-- Representation base (common attributes and elements) -->
  <xs:complexType name="RepresentationBaseType">
    <xs:sequence>
      <xs:element name="FramePacking" type="DescriptorType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="AudioChannelConfiguration" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="ContentProtection" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="EssentialProperty" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="SupplementalProperty" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="InbandEventStream" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="profiles" type="xs:string"/>
    <xs:attribute name="width" type="xs:unsignedInt"/>
    <xs:attribute name="height" type="xs:unsignedInt"/>
    <xs:attribute name="sar" type="RatioType"/>
    <xs:attribute name="frameRate" type="FrameRateType"/>
    <xs:attribute name="audioSamplingRate" type="xs:string"/>
    <xs:attribute name="mimeType" type="xs:string"/>
    <xs:attribute name="segmentProfiles" type="xs:string"/>
    <xs:attribute name="codecs" type="xs:string"/>
    <xs:attribute name="maximumSAPPeriod" type="xs:double"/>
    <xs:attribute name="startWithSAP" type="SAPType"/>
    <xs:attribute name="maxPlayoutRate" type="xs:double"/>
    <xs:attribute name="codingDependency" type="xs:boolean"/>
    <xs:attribute name="scanType" type="VideoScanType"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Stream Access Point type enumeration -->
  <xs:simpleType name="SAPType">
    <xs:restriction base="xs:unsignedInt">
      <xs:minInclusive value="0"/>
      <xs:maxInclusive value="6"/>
    </xs:restriction>
  </xs:simpleType>

  <!-- Video Scan type enumeration -->
  <xs:simpleType name="VideoScanType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="progressive"/>
      <xs:enumeration value="interlaced"/>
      <xs:enumeration value="unknown"/>
    </xs:restriction>
  </xs:simpleType>

  <!-- Subset  -->
  <xs:complexType name="SubsetType">
    <xs:attribute name="contains" type="UIntVectorType" use="required"/>
    <xs:attribute name="id" type="xs:string"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Segment information base -->
  <xs:complexType name="SegmentBaseType">
    <xs:sequence>
      <xs:element name="Initialization" type="URLType" minOccurs="0"/>
      <xs:element name="RepresentationIndex" type="URLType" minOccurs="0"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="timescale" type="xs:unsignedInt"/>
    <xs:attribute name="presentationTimeOffset" type="xs:unsignedLong"/>
    <xs:attribute name="indexRange" type="xs:string"/>
    <xs:attribute name="indexRangeExact" type="xs:boolean" default="false"/>
    <xs:attribute name="availabilityTimeOffset" type="xs:double"/>
    <xs:attribute name="availabilityTimeComplete" type="xs:boolean"/>
```

```
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Multiple Segment information base -->
  <xs:complexType name="MultipleSegmentBaseType">
    <xs:complexContent>
      <xs:extension base="SegmentBaseType">
        <xs:sequence>
          <xs:element name="SegmentTimeline" type="SegmentTimelineType" minOccurs="0"/>
          <xs:element name="BitstreamSwitching" type="URLType" minOccurs="0"/>
        </xs:sequence>
        <xs:attribute name="duration" type="xs:unsignedInt"/>
        <xs:attribute name="startNumber" type="xs:unsignedInt"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <!-- Segment Info item URL/range -->
  <xs:complexType name="URLType">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="sourceURL" type="xs:anyURI"/>
    <xs:attribute name="range" type="xs:string"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Segment List -->
  <xs:complexType name="SegmentListType">
    <xs:complexContent>
      <xs:extension base="MultipleSegmentBaseType">
        <xs:sequence>
          <xs:element name="SegmentURL" type="SegmentURLType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute ref="xlink:href"/>
        <xs:attribute ref="xlink:actuate" default="onRequest"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <!-- Segment URL  -->
  <xs:complexType name="SegmentURLType">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="media" type="xs:anyURI"/>
    <xs:attribute name="mediaRange" type="xs:string"/>
    <xs:attribute name="index" type="xs:anyURI"/>
    <xs:attribute name="indexRange" type="xs:string"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Segment Template -->
  <xs:complexType name="SegmentTemplateType">
    <xs:complexContent>
      <xs:extension base="MultipleSegmentBaseType">
        <xs:attribute name="media" type="xs:string"/>
        <xs:attribute name="index" type="xs:string"/>
        <xs:attribute name="initialization" type="xs:string"/>
        <xs:attribute name="bitstreamSwitching" type="xs:string"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <!-- Segment Timeline -->
  <xs:complexType name="SegmentTimelineType">
    <xs:sequence>
      <xs:element name="S" minOccurs="1" maxOccurs="unbounded">
        <xs:complexType>
          <xs:attribute name="t" type="xs:unsignedLong"/>
          <xs:attribute name="d" type="xs:unsignedLong" use="required"/>
          <xs:attribute name="r" type="xs:integer" use="optional" default="0"/>
          <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>
      </xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
```

```
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Whitespace-separated list of strings -->
  <xs:simpleType name="StringVectorType">
    <xs:list itemType="xs:string"/>
  </xs:simpleType>

  <!-- Whitespace-separated list of unsigned integers -->
  <xs:simpleType name="UIntVectorType">
    <xs:list itemType="xs:unsignedInt"/>
  </xs:simpleType>

  <!-- Base URL -->
  <xs:complexType name="BaseURLType">
    <xs:simpleContent>
      <xs:extension base="xs:anyURI">
        <xs:attribute name="serviceLocation" type="xs:string"/>
        <xs:attribute name="byteRange" type="xs:string"/>
        <xs:attribute name="availabilityTimeOffset" type="xs:double"/>
        <xs:attribute name="availabilityTimeComplete" type="xs:boolean"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

  <!-- Program Information -->
  <xs:complexType name="ProgramInformationType">
    <xs:sequence>
      <xs:element name="Title" type="xs:string" minOccurs="0"/>
      <xs:element name="Source" type="xs:string" minOccurs="0"/>
      <xs:element name="Copyright" type="xs:string" minOccurs="0"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="lang" type="xs:language"/>
    <xs:attribute name="moreInformationURL" type="xs:anyURI"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Descriptor -->
  <xs:complexType name="DescriptorType">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="schemeIdUri" type="xs:anyURI" use="required"/>
    <xs:attribute name="value" type="xs:string"/>
    <xs:attribute name="id" type="xs:string"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Metrics -->
  <xs:complexType name="MetricsType">
    <xs:sequence>
      <xs:element name="Reporting" type="DescriptorType" maxOccurs="unbounded"/>
      <xs:element name="Range" type="RangeType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="metrics" type="xs:string" use="required"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- Metrics Range -->
  <xs:complexType name="RangeType">
    <xs:attribute name="starttime" type="xs:duration"/>
    <xs:attribute name="duration" type="xs:duration"/>
  </xs:complexType>

</xs:schema>
```

Annex C
(normative)

MIME type registration for MPD

C.1 Introduction

This Annex provides the formal MIME type registration for the MPD. It is referenced from the registry at http://www.iana.org/.

C.2 MIME type and subtype

The MIME Type and Subtype are defined as follows:

— MIME media type name: application

— MIME subtype name: dash+xml

— Required parameters: none

— Optional parameters: The '`profiles`' parameter as documented in Annex C.3.1 in this part of ISO/IEC 23009

— Encoding considerations: UTF-8

— Security considerations: The MPD is a Media Presentation Description and contains references to other resources. It is coded in XML, and there are risks that deliberately malformed XML could cause security issues. In addition, an MPD could be authored that causes receiving clients to access other resources; if widely distributed, this could be used to cause a denial-of-service attack.

The Media Presentation Description (MPD) format does not incorporate any active or executable content. However, other forms of material from outside sources can be referenced by an MPD, and this material could contain active or executable content. Such material is expected to be identified by its own MIME type, and the security considerations of that format should be taken into account.

If operating in an insecure environment and required by the content/service provider, elements and attributes of MPD may be encrypted to protect their confidentiality by using the syntax and processing rules specified in the W3C Recommendation "XML Encryption Syntax and Processing".

If operating in an insecure environment and required by the content/service provider, the digital signing and verification procedures specified in the W3C Recommendation "XML Signature Syntax and Processing" may be used to protect data origin authenticity and integrity of the MPD.

— Interoperability considerations:
The specification defines a platform-independent expression of a presentation, and it is intended that wide interoperability can be achieved.

— Published specification: ISO/IEC 23009-1: Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats

— Applications which use this media type: various

— Additional information:

  — File extension(s): mpd

  — Intended usage: common

— Other Information/General Comment: none

— Person to contact for further information:

  — Name: Thomas Stockhammer

  — Email: stockhammer@nomor.de

— Author/Change controller: ISO/IEC JTC1/SC29 (MPEG)

C.3 Parameters

C.3.1 The profiles parameter

Parameter name: `profiles`

Parameter value: The 'profiles' parameter is an optional parameter that indicates one or more profiles to which the file claims conformance. The contents of this attribute shall conform to either the `pro-simple` or `pro-fancy` productions of RFC6381, Section 4.5. The profile identifiers reported in the MIME type parameter should match identically the profiles reported in the profiles attribute in the MPD itself (see Clause 8).

example:

```
application/dash+xml;profiles="urn:mpeg:dash:profile:full:2011,urn:3GPP:PSS
   :profile:DASH10"
```

C.4 MPD Anchors

C 4.1 General

URIs for resources with MIME type `application/dash+xml` may use URI fragment syntax to start a presentation at a given time and a given state.

An MPD anchor is a set of Representations being presented and a time offset from the start of a period on the media timeline. These are expressed using URI fragment syntax. This Annex defines one temporal parameter, position, and two context parameters, state and selection, in order to express the state of a DASH media presentation.

URI fragment starts with the '#' character, and is a string terminating the URI. MPD fragments shall be a comma-separated list of key=value pairs, with syntax and semantics of key and value parameters defined in Table C.1 of C.4.2.

C 4.2 Parameters

Table C.1 — Parameters for MPD Anchors

Key	Value	Semantics
`t`	Time or time range in the same format as defined in Media Fragments URI 1.0	Time since the beginning of the period indicated by the period parameter. If the `t` parameter is not present, its default value is 0.
`period`	String	`Period@id`. If period parameter is not present, the default value is the ID of the `Period` with the earliest *PeriodStart*.
`as`	string	Value of a single `AdaptationSet@id`
`track`	string	Value of a single `AdaptationSet@group`

NOTE 1: percent coding, defined in RFC 5986 needs to be used for all reserved characters in parameter values.

NOTE 2: ability to address elements in the MPD depends on the `Period@id`, `AdaptationSet@id` and `AdaptationSet@group`. Hence MPD authors should put these attributes explicitly into the MPD if they intend to make MPDs addressable.

C 4.3 Examples

42nd minute of Period1

```
my.mpd# t=42&period=Period1
```

42nd minute from the start of the presentation, English 5.1 audio and video

```
my.mpd#t=42&track=en51&track=vid
```

Set of Adaptation Sets from starting from the start of Period1

```
my.mpd#period=Period1&group=123
```

Annex D
(normative)

DASH Metrics

D.1 Introduction

This Annex defines the ISO/IEC 23009-1 DASH Metrics. The normative aspects of the Annex are defined in D.4, namely the semantics of the metrics and the associated keys to be used for requesting the collection of the metrics. The client reference model in D.2 and the observation points in D.3 serve as informative background information.

D.2 DASH-Metrics client reference model

The DASH-Metrics client reference model is depicted in highlighting so-called observation points (OPs) as defined in D.3.

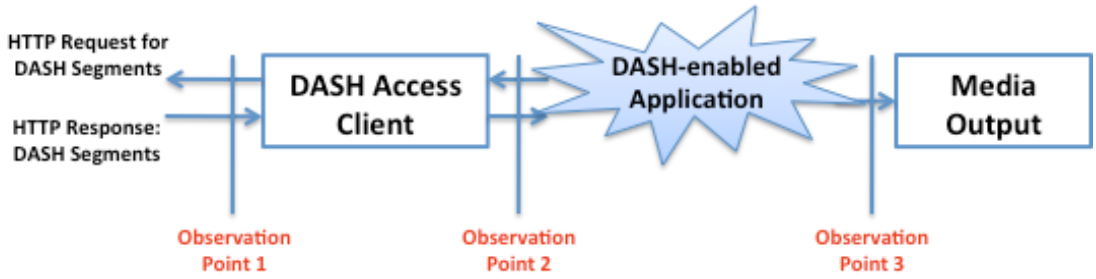

Figure D.1 — DASH-Metrics client reference model

The *DASH access client* as defined in 4.2 in this part of ISO/IEC 23009, issues HTTP requests (for DASH data structures), and receives HTTP request responses (containing DASH data structures). Data structures may typically be MPDs, Segments or partial Segments. This input/output interface from the network towards the DASH client is referred to as observation point 1 (OP1).

Furthermore, the DASH client delivers encoded media samples to the *DASH-enabled application* for further processing and may receive also commands from it. This input/output interface of the DASH client towards the DASH-enabled application is referred to as observation point 2 (OP2).

NOTE Further processing may include de-multiplexing (of audio/video) and/or decoding potentially involving several buffers.

Finally, the DASH-enabled application delivers decoded media samples to the *media output*, which displays the media to the user. This output interface towards the user is referred to as observation point 3 (OP3).

D.3 Definition of observation points

D.3.1 Introduction

This Clause defines the observation points as depicted in Figure D.1.

D.3.2 Observation point 1

The observation point 1 (OP1) is defined as:

— a set of TCP connections each defined by its destination IP address, initiation, connect and close times;

— a sequence of transmitted HTTP requests, each defined by its transmission time, contents, and the TCP connection on which it is sent; and

— for each HTTP response, the reception time and contents of the response header and the reception time of each byte of the response body.

NOTE The contents of the response body is fully defined by the contents of the request and response headers.

D.3.3 Observation point 2

The observation point 2 (OP2) consists of encoded media samples. Each encoded media sample is defined as:

— media type;

— decoding time;

— presentation time;

— the `@id` of the Representation from which the sample is taken; and

— the delivery time.

D.3.4 Observation point 3

The observation point 3 (OP3) consists of decoded media samples. Each decoded media sample is defined as:

— the media type;

— the presentation timestamp of the sample (media time);

— the actual presentation time of the sample (real time); and

— the `@id` of the Representation from which the sample is taken (the highest dependency level if the sample was constructed from multiple Representations).

D.4 Semantics of the DASH metrics

D.4.1 Introduction

This subclause provides the general QoE metric definitions and measurement framework.

The semantics are defined using an abstract syntax. Items in this abstract syntax have one of the following primitive types (`Integer`, `Real`, `Boolean`, `Enum`, `String`) or one of the following compound types:

— `Objects`: an unordered sequence of (`key`, `value`) pairs, where the key always has string type and is unique within the sequence.

— `List`: an ordered list of items.

— `Set`: an unordered set of items.

Additionally, there are two kinds of timestamp defined, i.e., *real time* (wall-clock time) with type `Real-Time` and *media time* with type `Media-Time`.

Where lists are defined the name '*`entry`*' is used to define the format of each entry, but since lists contain unnamed entries this name would not appear in any concrete syntax.

Each metric is defined as a named list of entries that logically contains the metric information for the entire Media Presentation. Reporting of these lists, whether done at the end of the Media Presentation or incrementally during the Media Presentation is out of scope of this specification.

D.4.2 TCP connections

Table D.1 contains the metric defining the list of TCP connections. The key in Table D.1 shall be used to refer to the metric as defined in Table D.1.

Table D.1 — List of TCP connections

Key			Type	Description
`TcpList`			`List`	List of HTTP request/response transactions
	Entry		`Object`	An entry for a single HTTP request/response
		`tcpid`	`Integer`	Identifier of the TCP connection on which the HTTP request was sent.
		`dest`	`String`	IP Address of the interface over which the client is receiving the TCP data.
		`topen`	`Real-Time`	The time at which the connection was opened (sending time of the initial SYN or connect socket operation).
		`tclose`	`Real-Time`	The time at which the connection was closed (sending or reception time of FIN or RST or close socket operation).
		`tconnect`	`Integer`	Connect time in ms (time from sending the initial SYN to receiving the ACK or completion of the connect socket operation).

D.4.3 HTTP request/response transactions

Table D.2 contains the metric defining the List of HTTP Request/Response Transactions. The key in Table D.2 shall be used to refer to the metric as defined in Table D.2.

Table D.2 — List of HTTP request/response transactions

Key	Type	Description
HttpList	List	List of HTTP request/response transactions
Entry	Object	An entry for a single HTTP request/response
tcpid	Integer	Identifier of the TCP connection on which the HTTP request was sent.
type	Enum	This is an optional parameter and should not be included in HTTP request/response transactions for progressive download. The type of the request: - MPD - XLink expansion - Initialization Segment - Index Segment - Media Segment - Bitstream Switching Segment - other
url	String	The original URL (before any redirects or failures)
actualurl	String	The actual URL requested, if different from above
range	String	The contents of the byte-range-spec part of the HTTP Range header.
trequest	Real-Time	The real time at which the request was sent.
tresponse	Real-Time	The real time at which the first byte of the response was received.
responsecode	Integer	The HTTP response code.
interval	Integer	The duration of the throughput trace intervals (ms), for successful requests only.
trace	List	Throughput trace, for successful requests only.
Entry	Object	A single throughput measurement entry.
s	Real-Time	Measurement period start.
d	Integer	Measurement period duration (ms).
b	List	List of integers counting the bytes received in each trace interval within the measurement period.

NOTE

1) Information additional to that specified in the type may be returned, for example if a client makes a request for a initialization information from a self-initializing Media Segment then Segment Index may also be received.

2) All entries for a given object will have the same URL and range and so can easily be correlated. If there were redirects or failures there will be one entry for each redirect/failure. The redirect-to URL or alternative `url` (where multiple have been provided in the MPD) will appear as the `actualurl` of the next entry with the same `url` value.

3) The periods in *Entry* should be those periods where the client was actively reading from the TCP connections (i.e., they should not include periods where the TCP connection is idle due to zero receive window).

The end of the last measurement period in the `trace` shall be the time at which the last byte of the response was received.

The `interval` and `trace` shall be absent for redirect and failure records.

The key `HttpList(`*`n,type`*`)` where *n* is a positive integer is defined for an HttpList with an interval of *n* ms and *type* is one of `MPD`, `XLinkExpansion`, `InitializationSegment`, `MediaSegment`, `IndexSegment` `BitstreamSwitchingSegment` or `other`. If *type* is not present, all HTTP transactions are requested to be collected. If *type* is present, it specifies that the HTTP transactions concerning a resource equal to *type* are requested to be collected. Multiple keys `HttpList`(*n,type*) with different values of *n* and *type* may be present for a single `@metrics` attribute value.

An HTTP transaction that is not finished within a QoE metric collection period shall not be included in the metrics.

D.4.4 Representation switch events

Table D.3 defines the metric for Representation switch events. The key in Table D.3 shall be used to refer to the metric as defined in Table D.3.

Table D.3 — List of Representation switch events

Key			Type	Description
`RepSwitchList`			`List`	List of Representation switch events (a switch event is the time at which the first HTTP request for a new Representation, that is later presented, is sent)
	Entry		`Object`	A Representation switch event.
		`t`	`Real-Time`	Time of the switch event.
		`mt`	`Media-Time`	The media presentation time of the earliest access unit (out of all media content components) played out from the “to” Representation.
		`to`	`String`	value of **`Representation`**`@id` identifying the switch-to Representation.
		`lto`	`Integer`	If not present, this metrics concerns the Representation as a whole. If present, `lto` indicates the value of **`SubRepresentation`**`@level` within Representation identifying the switch-to level of the Representation.

D.4.5 Buffer level

Table D.4 defines the metric for buffer level status events. The key in Table D.4 shall be used to refer to the metric as defined in Table D.4.

Table D.4 — List of buffer level

Key	Type	Description
`BufferLevel`	`List`	List of buffer occupancy level measurements during playout at normal speed.
Entry	`Object`	One buffer level measurement.
`t`	`Real-Time`	Time of the measurement of the buffer level.
`level`	`Integer`	Level of the buffer in milliseconds. Indicates the playout duration for which media data of all active media components is available starting from the current playout time.

The key is `BufferLevel(`*n*`)`, where *n* is a positive integer is defined to refer to the metric in which the buffer level is recorded every *n* ms.

D.4.6 Play list

Decoded samples are generally rendered in presentation time sequence, each at or close to its specified presentation time. A compact Representation of the information flow can thus be constructed from a list of time periods during which samples of a single Representation were continuously rendered, such that each was presented at its specified presentation time to some specific level of accuracy (e.g., +/-10ms).

Such a sequence of periods of continuous delivery is started by a user action that requests playout to begin at a specified media time (this could be a “play”, “seek” or “resume” action) and continues until playout stops either due to a user action, the end of the content, or a permanent failure.

Table D.5 defines the play list event metric. The key in Table D.5 shall be used to refer to the metric as defined in Table D.5.

Table D.5 — Play list

Key	Type	Description
`PlayList`	`List`	A list of playback periods. A playback period is the time interval between a user action and whichever occurs soonest of the next user action, the end of playback or a failure that stops playback.
Entry	`Object`	A record of a single playback period.
`start`	`Real-Time`	Timestamp of the user action that starts the playback period.
`mstart`	`Media-Time`	The presentation time at which playout was requested by the user action.
`starttype`	`Enum`	Type of user action which triggered playout - New playout request (e.g. initial playout or seeking)

Key	Type	Description
		- Resume from pause - Other user request (e.g. user-requested quality change) - Start of a metrics collection period (hence earlier entries in the play list not collected)
trace	List	List of periods of continuous rendering of decoded samples.
Entry	Objects	Single entry in the list.
representationid	String	The value of the **`Representation`**`@id` of the Representation from which the samples were taken.
subreplevel	Integer	If not present, this metrics concerns the Representation as a whole. If present, **`subreplevel`** indicates the greatest value of any **`Subrepresentation`**`@level` being rendered.
start	Real-Time	The time at which the first sample was rendered.
mstart	Media-Time	The presentation time of the first sample rendered.
duration	Integer	The duration of the continuously presented samples (which is the same in real time and media time). “Continuously presented” means that the media clock continued to advance at the playout speed throughout the interval.
playbackspeed	Real	The playback speed relative to normal playback speed (i.e.normal forward playback speed is 1.0).
stopreason	Enum	The reason why continuous presentation of this Representation was stopped. Either: - Representation switch (not relevant in case of progressive download) - rebuffering - user request - end of Period - end of content - end of a metrics collection period - failure

NOTE The trace may include entries for different representations that overlap in time, because multiple representations are being rendered simultaneously, for example one audio and one video Representation.

Annex E
(normative)

Byte range requests with regular HTTP GET methods

E.1 Background

There exist deployment environments where HTTP partial GET is not supported, or results in the return of the entire, rather than partial target. This represents a problem for DASH clients. It is expected that these problems gradually disappear, but until this will be the case, a method is provided to not exclude DASH clients operating in this environments and service providers wanting to support such clients, are excluded from using this DASH standard to deploy media streaming services using the formats defined in this part of ISO/IEC 23009. Still it is expected that this Annex will be deprecated in some later versions of this standard.

To address these requirements the **BaseURL**@byteRange attribute may be present. If present, it provides information that resources offered in the MPD but are requested by a HTTP partial GET (e.g. Segments for which HTTP-URLs contain byte ranges or Subsegments) may also be requested using a regular HTTP GET by mapping the information that is otherwise added in the Range header into the request URI of a regular HTTP GET request. It is expected that DASH clients only use this method if HTTP partial GET requests fail. If DASH clients only have this alternative to request segments or Subsegments, then it is expected that they request single units of segments or Subsegments.

E.2 Construction rule

The **BaseURL**@byteRange attribute represents a template that may be used to construct a URL requesting a byte range (a "byte range URL") from a resource, given the original URL of the resource and the required byte range. The result of issuing a GET request to this byte range URL without including the HTTP Range header should be identical to the result of requesting the original URL with the byte range specified in the HTTP Range header.

The **BaseURL**@byteRange contains a template string that contains one or more of the identifiers as listed in Table E.1. The string shall contain identifiers $first$ and $last$ as specified in Table E.1.

The byte range URL shall be constructed from the template string by substituting the identifiers specified in the first column Table E.1 with the values specified in the second column of Table E.1. If the $query$ identifier is not present in the template and the query portion of the original URL as defined in RFC 3986 is not empty, then the string "?" query shall be appended to the constructed URL.

If the template string contains unrecognized identifiers then the result of the URL construction is unspecified. In this case it is expected that the DASH Client ignores the entire containing **ByteRange** element and the processing of the MPD continues as if this **ByteRange** element was not present.

Strings outside identifiers shall only contain characters that permit to form a valid HTTP-URL according to RFC 3986.

Table E.1 — Identifiers for Byte Range Templates

$<Identifier>$	Substitution parameter
$$	Is an escape sequence, i.e. "$$" is replaced with a single "$"
$base$	The identifier shall be substituted by the scheme ":" hier-part of the original URL as defined in RFC3986.
$query$	The identifier shall be substituted by the query part of the original URL as defined in RFC3986. If the query part of the original URL is empty then inclusion of this identifier in the template shall cause removal of the separator character immediately preceding the $query$ identifier in the template string if that character is not the "?" character, or, otherwise, the separator character immediately following the $query$ identifier if present.
$first$	The identifier shall be substituted by the byte offset of the first byte in a range and shall be identical to the value of 'first-byte-pos' of 'byte-range-spec' in 14.35.1 of RFC2616, if this request would be executed using a partial GET request.
$last$	The identifier is substituted by the byte offset of the last byte in the range; that is, the byte positions specified are inclusive. It shall be identical to the value of 'last-byte-pos' of 'byte-range-spec' in 14.35.1 of RFC2616, if this request would be executed using a partial GET request.

E.3 Examples

Original URL	http://cdn.example.com/movies/134532/audio/en/aac64.mp4?token=8787r08f2gf087g28gf926
Byte Range	1876-23456
BaseURL@byteRange	$base$/range/$first$-$last$
Byte range URL	http://cdn.example.com/movies/134532/audio/en/aac64.mp4/range/1876-23456?token=8787r08f2gf087g28gf926

Original URL	http://cdn.example.com/movies/134532/audio/en/aac64.mp4
Byte Range	1876-23456
BaseURL@byteRange	$base$?$query$&range=$first$-$last$
Byte range URL	http://cdn.example.com/movies/134532/audio/en/aac64.mp4?**range=1876-23456**

Original URL	http://cdn.example.com/movies/134532/audio/en/aac64.mp4?token=8787r08f2gf087g28gf926
Byte Range	1876-23456
BaseURL@byteRange	$base$?$query$&range=$first$-$last$
Byte range URL	http://cdn.example.com/movies/134532/audio/en/aac64.mp4?token=8787r08f2gf087g28gf926**&range=1876-23456**

Annex F
(informative)

Guidelines for extending DASH with other delivery formats

F.1 Adding delivery formats to DASH

In order to support use with DASH, a delivery format should have the property that decoding and playback of any portion of the media can be achieved using a subset of the media that is only a constant amount larger than the portion of the media to be played.

For example, a delivery format where the media is stored as a header followed by a sequence of small blocks, with the property that any block can be decoded and played out given only that block and the header has this property. The definition of these blocks and the mapping to the Subsegments in this part of ISO/IEC 23009 is encouraged. A Subsegment may be defined as a contiguous time interval of a Segment and a contiguous byte range of a Segment, for which no overlap in both dimensions with any other Subsegment in the Segment exists.

Additionally, it is desirable that the delivery format supports some kind of "index" which enables the byte range within the Segment corresponding to any given time range to be efficiently discovered. A suitable unit is the indexing of Subsegments. It should be possible to discover the position in the Segment of the index without downloading the whole Segment. The position of the index may also be advertised in the MPD or the index may be provided as a separate Index Segment. The Segment Index ('`sidx`') or Subsegment Index ('`ssix`'), both defined in ISO/IEC 14496-12 may serve as a starting point and/or may be directly applied to any other media format.

F.2 Media Presentation authoring rules

A specification for how to use a media container format with DASH should include:

- Definition of the MIME type for the Representation as a concatenation of Segments.
- Description of either a self-initializing Media Segment or the combination of an Initialization Segment and a Media Segment format.

In addition, the specification may further define:

- Index Segments
- Bitstream switching segments
- Interpretation of a media Stream Access Point (SAP), potentially different types as defined in 4.5.2 in the context of the media container format. (The SAP types are fully *defined* in Annex I of ISO/IEC 14496-12 and should not be re-defined, but the *interpretation* of those definitions in media-container-specific language may be necessary).
- Container-format-specific semantics for the `@bitstreamSwitching`, `@segmentAlignment` and `@subsegmentAlignment`. These should align with the definitions in this part of ISO/IEC 23009.

Note that Representation attributes present in the MPD may also be repeated in the media itself, e.g. in an Initialization Segment or a Media Segment. The media content should be provided such that no mismatch between these two values occurs. If it does, the value in the media itself is expected to take precedence over values expressed in the MPD, especially when used in the media decoding process.

Annex G
(informative)

MPD Examples and MPD Usage

G.1 Example MPD for ISO Base media file format On Demand profile

This subclause provides a simple example for a static presentation with self-initializing Media Segments, multiple languages, subtitles, content protection and multiple base URLs. This MPD document describes content available from two sources (cdn1 and cdn2) that has audio available in English or French at rates of 64kbits and 32kbits and subtitles in German. Six versions of the video are provided at bitrates between 256kbit/s and 2Mbit/s in different spatial resolutions. Content protection is applied.

The Media Presentation complies with the ISO Base media file format On Demand profile as defined in 8.3.

```
<?xml version="1.0" encoding="UTF-8"?>
<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="static"
  mediaPresentationDuration="PT3256S"
  minBufferTime="PT1.2S"
  profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">

  <BaseURL>http://cdn1.example.com/</BaseURL>
  <BaseURL>http://cdn2.example.com/</BaseURL>

  <Period>
    <!-- English Audio -->
    <AdaptationSet mimeType="audio/mp4" codecs="mp4a.40" lang="en" subsegmentAlignment="true"
subsegmentStartsWithSAP="1">
      <ContentProtection schemeIdUri="urn:uuid:706D6953-656C-5244-4D48-656164657221"/>
      <Representation id="1" bandwidth="64000">
        <BaseURL>7657412348.mp4</BaseURL>
      </Representation>
      <Representation id="2" bandwidth="32000">
        <BaseURL>3463646346.mp4</BaseURL>
      </Representation>
    </AdaptationSet>
    <!-- French Audio -->
    <AdaptationSet mimeType="audio/mp4" codecs="mp4a.40.2" lang="fr" subsegmentAlignment="true"
subsegmentStartsWithSAP="1">
      <ContentProtection schemeIdUri="urn:uuid:706D6953-656C-5244-4D48-656164657221"/>
      <Role schemeIdUri="urn:mpeg:dash:role:2011" value="dub"/>
      <Representation id="3" bandwidth="64000">
        <BaseURL>3463275477.mp4</BaseURL>
      </Representation>
      <Representation id="4" bandwidth="32000">
        <BaseURL>5685763463.mp4</BaseURL>
      </Representation>
    </AdaptationSet>
    <!-- Timed text -->
    <AdaptationSet mimeType="application/ttml+xml" lang="de">
      <Role schemeIdUri="urn:mpeg:dash:role" value="subtitle"/>
      <Representation id="5" bandwidth="256">
        <BaseURL>796735657.xml</BaseURL>
      </Representation>
    </AdaptationSet>
    <!-- Video -->
    <AdaptationSet mimeType="video/mp4" codecs="avc1.4d0228" subsegmentAlignment="true"
subsegmentStartsWithSAP="2">
      <ContentProtection schemeIdUri="urn:uuid:706D6953-656C-5244-4D48-656164657221"/>
      <Representation id="6" bandwidth="256000" width="320" height="240">
        <BaseURL>8563456473.mp4</BaseURL>
      </Representation>
```

```
      <Representation id="7" bandwidth="512000" width="320" height="240">
        <BaseURL>56363634.mp4</BaseURL>
      </Representation>
      <Representation id="8" bandwidth="1024000" width="640" height="480">
        <BaseURL>562465736.mp4</BaseURL>
      </Representation>
      <Representation id="9" bandwidth="1384000" width="640" height="480">
        <BaseURL>41325645.mp4</BaseURL>
      </Representation>
      <Representation id="A" bandwidth="1536000" width="1280" height="720">
        <BaseURL>89045625.mp4</BaseURL>
      </Representation>
      <Representation id="B" bandwidth="2048000" width="1280" height="720">
        <BaseURL>23536745734.mp4</BaseURL>
      </Representation>
    </AdaptationSet>
  </Period>
</MPD>
```

G.2 Example for ISO Base media file format Live profile

This subclause provides a simple example for a dynamic presentation, with multiple languages, multiple base URLs, multiple video bitrates, and segments about two seconds in length for low latency from live programming. At the time this MPD was fetched, 432 Segments of the dynamic presentation were available so the wall clock time must have been approximately 2011-12-25T12:44:24 UTC. All the video Segments are aligned and start with a Stream Access Point. All the audio Segments are aligned so language switching can done with the non-language sound (e.g. music) seamlessly.

In this MPD assuming that the first **`BaseURL`** element and the video Representation with id "`v1`" is selected, and template results in `http://cdn1.example.com/video/50000/$Time$.mp4v`, the segment list starting at number 0 results in

```
http://cdn1.example.com/video/500000/0.mp4v
http://cdn1.example.com/video/500000/180180.mp4v
http://cdn1.example.com/video/500000/360360.mp4v
http://cdn1.example.com/video/500000/540540.mp4v
http://cdn1.example.com/video/500000/720720.mp4v
...
```

The Media Presentation conforms to the ISO Base media file format Live profile in 8.4.

```
<?xml version="1.0" encoding="UTF-8"?>
<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="PT30M"
  availabilityStartTime="2011-12-25T12:30:00"
  minBufferTime="PT4S"
  profiles="urn:mpeg:dash:profile:isoff-live:2011">

  <BaseURL>http://cdn1.example.com/</BaseURL>
  <BaseURL>http://cdn2.example.com/</BaseURL>

  <Period id="1">
    <!-- Video -->
    <AdaptationSet
      mimeType="video/mp4"
      codecs="avc1.4D401F"
      frameRate="30000/1001"
      segmentAlignment="true"
      startWithSAP="1">
      <BaseURL>video/</BaseURL>
      <SegmentTemplate timescale="90000" initialization="$Bandwidth%/init.mp4v"
```

```
media="$Bandwidth%/$Time$.mp4v">
        <SegmentTimeline>
          <S t="0" d="180180" r="432"/>
        </SegmentTimeline>
      </SegmentTemplate>
      <Representation id="v0" width="320" height="240" bandwidth="250000"/>
      <Representation id="v1" width="640" height="480" bandwidth="500000"/>
      <Representation id="v2" width="960" height="720" bandwidth="1000000"/>
    </AdaptationSet>
    <!-- English Audio -->
    <AdaptationSet mimeType="audio/mp4" codecs="mp4a.40" lang="en" segmentAlignment="0"
startWithSAP="1">
      <SegmentTemplate timescale="48000" initialization="audio/en/init.mp4a"
media="audio/en/$Time$.mp4a">
        <SegmentTimeline>
          <S t="0" d="96000" r="432"/>
        </SegmentTimeline>
      </SegmentTemplate>
      <Representation id="a0" bandwidth="64000" />
    </AdaptationSet>
    <!-- French Audio -->
    <AdaptationSet mimeType="audio/mp4" codecs="mp4a.40" lang="fr" segmentAlignment="0"
startWithSAP="1">
      <SegmentTemplate timescale="48000" initialization="audio/fr/init.mp4a"
media="audio/fr/$Time$.mp4a">
        <SegmentTimeline>
          <S t="0" d="96000" r="432"/>
        </SegmentTimeline>
      </SegmentTemplate>
      <Representation id="a0" bandwidth="64000" />
    </AdaptationSet>
  </Period>
</MPD>
```

G.3 Example for MPEG-2 TS Simple profile

This subclause introduces a simple example for a static presentation, with multiple languages, multiple base URLs, multiple video bitrates, and segments about four seconds in length.

In this MPD assuming that the first **BaseURL** element and the Representation with id "1400kbps" is selected, and template results in `http://cdn1.example.com/SomeMovie_1400kbps_$Number%05$.ts`, the segment list starting at number 0 results in

```
http://cdn1.example.com/SomeMovie_1400kbps_00001.ts
http://cdn1.example.com/SomeMovie_1400kbps_00002.ts
http://cdn1.example.com/SomeMovie_1400kbps_00003.ts
http://cdn1.example.com/SomeMovie_1400kbps_00004.ts
http://cdn1.example.com/SomeMovie_1400kbps_00005.ts
...
```

The Media Presentation conforms to the profile in 8.7.

```
<?xml version="1.0" encoding="UTF-8"?>
<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="static"
  mediaPresentationDuration="PT6158S"
  availabilityStartTime="2011-05-10T06:16:42"
  minBufferTime="PT1.4S"
  profiles="urn:mpeg:dash:profile:mp2t-simple:2011"
  maxSegmentDuration="PT4S">

  <BaseURL>http://cdn1.example.com/</BaseURL>
  <BaseURL>http://cdn2.example.com/</BaseURL>
```

```
  <Period id="42" duration="PT6158S">
    <AdaptationSet
      mimeType="video/mp2t"
      codecs="avc1.4D401F,mp4a"
      frameRate="24000/1001"
      segmentAlignment="true"
      subsegmentAlignment="true"
      bitstreamSwitching="true"
      startWithSAP="2"
      subsegmentStartsWithSAP="2">
      <ContentComponent contentType="video" id="481"/>
      <ContentComponent contentType="audio" id="482" lang="en"/>
      <ContentComponent contentType="audio" id="483" lang="es"/>
      <BaseURL>SomeMovie/</BaseURL>
      <SegmentTemplate
        media="$RepresentationID$ $Number%05d$.ts"
        index="$RepresentationID$.sidx"
        initialization="$RepresentationID$-init.ts"
        bitstreamSwitching="$RepresentationID$-bssw.ts"
        duration="4"/>
      <Representation id="720kbps" bandwidth="792000" width="640" height="368"/>
      <Representation id="1130kbps" bandwidth="1243000" width="704" height="400"/>
      <Representation id="1400kbps" bandwidth="1540000" width="960" height="544"/>
      <Representation id="2100kbps" bandwidth="2310000" width="1120" height="640"/>
      <Representation id="2700kbps" bandwidth="2970000" width="1280" height="720"/>
      <Representation id="3400kbps" bandwidth="3740000" width="1280" height="720"/>
    </AdaptationSet>
  </Period>
</MPD>
```

G.4 Example for multiple stereo views

This subclause introduces a simple example for a stereo video presentation from three cameras in one line where one stereo view is from the left-hand two cameras and the second is from the right-hand two cameras.

```
<?xml version="1.0" encoding="UTF-8"?>
<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="static"
  mediaPresentationDuration="PT3256S"
  minBufferTime="PT10.00S"
  profiles="urn:mpeg:dash:profile:isoff-main:2011">

  <BaseURL>http://www.example.com/</BaseURL>

  <!-- In this Period there are 3 views: coming from three lined up cameras: C1-C2-C3.
       C1+C2 and C2+C3 each form a stereo pair but C1+C3 does not.
       C2 is taken as the base view for MVC while C1 and C3 are enhancement views -->
  <Period start="PT0.00S" duration="PT2000.00S">
    <SegmentList>
      <Initialization sourceURL="seg-m-init.mp4"/>
    </SegmentList>
    <AdaptationSet mimeType="video/mp4" codecs="avc1.640828">
      <Role schemeIdUri="urn:mpeg:dash:stereoid:2011" value="l1 r0"/>
      <Representation id="C2" bandwidth="128000">
        <SegmentList duration="10">
          <SegmentURL media="seg-m1-C2view-1.mp4"/>
          <SegmentURL media="seg-m1-C2view-2.mp4"/>
          <SegmentURL media="seg-m1-C2view-3.mp4"/>
        </SegmentList>
      </Representation>
    </AdaptationSet>
    <!-- The following Adaptation set contains a Representation functionally identical
         to the Representation in the previous Adaptation set. Therefore, these both
         have the same Representation@id. This is done for compatibility to 2D receivers
         that do not understand the schemeIdURI of the Role Descriptor and may ignore the
         Adaptation set -->
    <AdaptationSet mimeType="video/mp4" codecs="avc1.640828">
      <Representation id="C2" bandwidth="128000">
```

```
        <SegmentList duration="10">
          <SegmentURL media="seg-m1-C2view-1.mp4"/>
          <SegmentURL media="seg-m1-C2view-2.mp4"/>
          <SegmentURL media="seg-m1-C2view-3.mp4"/>
        </SegmentList>
      </Representation>
    </AdaptationSet>
    <AdaptationSet mimeType="video/mp4" codecs="mvc1.760028">
      <Role schemeIdUri="urn:mpeg:dash:stereoid:2011" value="l0"/>
      <Representation id="C1" dependencyId="C2" bandwidth="192000">
        <SegmentList duration="10">
          <SegmentURL media="seg-m1-C1view-1.mp4"/>
          <SegmentURL media="seg-m1-C1view-2.mp4"/>
          <SegmentURL media="seg-m1-C1view-3.mp4"/>
        </SegmentList>
      </Representation>
    </AdaptationSet>
    <AdaptationSet mimeType="video/mp4" codecs="mvc1.760028">
      <Role schemeIdUri="urn:mpeg:dash:stereoid:2011" value="r1"/>
      <Representation id="C3" dependencyId="C2" bandwidth="192000">
        <SegmentList duration="10">
          <SegmentURL media="seg-m1-C3view-1.mp4"/>
          <SegmentURL media="seg-m1-C3view-2.mp4"/>
          <SegmentURL media="seg-m1-C3view-3.mp4"/>
        </SegmentList>
      </Representation>
    </AdaptationSet>
  </Period>

  <!--  In this Period there are only 2 views:  C1+C2 form a stereo pair;
        C2 is the base view for MVC and C1 is the enhancement view -->
  <Period duration="PT1256.00S">
    <SegmentList>
      <Initialization sourceURL="seg-m-init-2.mp4"/>
    </SegmentList>
    <AdaptationSet mimeType="video/mp4" codecs="avc1.640828">
      <Role schemeIdUri="urn:mpeg:dash:stereoid:2011" value="r0"/>
      <Representation id="C2" bandwidth="128000">
        <SegmentList duration="10">
          <SegmentURL media="seg-m1-C2view-201.mp4"/>
          <SegmentURL media="seg-m1-C2view-202.mp4"/>
        </SegmentList>
      </Representation>
    </AdaptationSet>
    <AdaptationSet mimeType="video/mp4" codecs=" mvc1.760028">
      <Role schemeIdUri="urn:mpeg:dash:stereoid:2011" value="l0"/>
      <Representation id="C1" dependencyId="C2" bandwidth="192000">
        <SegmentList duration="10">
          <SegmentURL media="seg-m1-C1view-201.mp4"/>
          <SegmentURL media="seg-m1-C1view-202.mp4"/>
        </SegmentList>
      </Representation>
    </AdaptationSet>
  </Period>
</MPD>
```

G.5 Example for SVC alternative streams

This simple example introduces a piece of SVC content split into three Representations with each additional bitrate depending on the previous ones.

```
<?xml version="1.0" encoding="UTF-8"?>
<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="static"
  mediaPresentationDuration="PT3256S"
  minBufferTime="PT1.2S"
  profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">
```

```
  <BaseURL>http://cdn1.example.com/</BaseURL>
  <BaseURL>http://cdn2.example.com/</BaseURL>

  <!-- In this Period the SVC stream is split into three representations -->
  <Period>
    <AdaptationSet
      subsegmentAlignment="true"
      subsegmentStartsWithSAP="2"
      minBandwidth="512000"
      maxBandwidth="1024000"
      width="640"
      height="480"
      frameRate="30"
      lang="en">
      <!-- Independent Representation -->
      <Representation
        mimeType="video/mp4"
        codecs="avc1.4D401E,mp4a.40"
        id="tag5"
        bandwidth="512000">
        <BaseURL>video-512k.mp4</BaseURL>
        <SegmentBase indexRange="0-4332"/>
      </Representation>
      <!-- Representation dependent on above -->
      <Representation
        mimeType="video/mp4"
        codecs="avc2.56401E"
        id="tag6"
        dependencyId="tag5"
        bandwidth="768000">
        <BaseURL>video-768k.mp4</BaseURL>
        <SegmentBase indexRange="0-3752"/>
      </Representation>
      <!-- Representation dependent on both above -->
      <Representation
        mimeType="video/mp4"
        codecs="avc2.56401E "
        id="tag7"
        dependencyId="tag5 tag6"
        bandwidth="1024000">
        <BaseURL>video-1024k.mp4</BaseURL>
        <SegmentBase indexRange="0-3752"/>
      </Representation>
    </AdaptationSet>
  </Period>
</MPD>
```

G.6 Example for trick play support

This subclause introduces a simple example for using Sub-Representations to support layered coding.

```
<?xml version="1.0" encoding="UTF-8"?>
<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="static"
  mediaPresentationDuration="PT3256S"
  minBufferTime="PT1.2S"
  profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">

  <BaseURL>http://cdn1.example.com/</BaseURL>
  <BaseURL>http://cdn2.example.com/</BaseURL>

  <!-- Period with a multiplexed stream with subrepresentations
       described for use with fast forward -->
  <Period>
    <AdaptationSet
      mimeType="video/mp4"  codecs="avc2.4D401E,avc1.4D401E,mp4a.40"
      width="640" height="480" frameRate="30" lang="en"
      subsegmentAlignment="true" subsegmentStartsWithSAP="2">
      <ContentComponent id="0"  contentType="video"/>
```

```
      <ContentComponent id="1"  contentType="audio"/>
      <Representation id="tag0" bandwidth="512000">
        <BaseURL>video-512k.mp4</BaseURL>
        <SubRepresentation level="0" contentComponent="0" bandwidth="128000" codecs="avc1.4D401E"
maxPlayoutRate="4"/>
        <SubRepresentation level="1" dependencyLevel="0" contentComponent="0" bandwidth="320000"
codecs="avc2.4D401E"/>
        <SubRepresentation level="2" contentComponent="1" bandwidth="64000" codecs="mp4a.40"/>
        <SegmentBase indexRange="7632" />
      </Representation>
    </AdaptationSet>
  </Period>
</MPD>
```

G.7 Example for content protected by multiple schemes

In the example below, *example.com* is a provider of CDN services and also a hosting service for movie service providers *MoviesSP*. The English audio and the video tracks are encrypted and licensed by *MoviesSP*. However, the French audio track is encrypted and licensed by a different service provider.

A hypothetical DRM standardization organization has registered a Scheme Type 'ZZZZ' with MP4REG and documented how scheme specific licensing information is stored entirely within the content so there is no additional information provided in the **ContentProtection** element. Since the scheme type is registered and the rules for its use are documented, the "urn:mpeg:dash:mp4protection" is used for the @schemeIdUri and "ZZZZ" is the assigned @value.

In addition, a second DRM scheme is used that comes from a DRM vendor who has published documentation of their system that declares that they use the DASH **ContentProtection** element with a @schemeIdUri attribute value "http://example.net/052011/drm". (This DRM vendor owns the domain *example.net* as of May, 2011.) Documentation for this scheme states that there must always be two URLs in the **ContentProtection** element that are placed in elements defined in the http://example.net/052011/drm namespace. The **License** element contains a license token and the **Content** element contains a content token. Regardless of which service provider uses the protection product from this DRM vendor, these rules must always be followed.

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="urn:mpeg:dash:schema:mpd:2011"
  xmlns:drm="http://example.net/052011/drm"
  xsi:schemaLocation=" urn:mpeg:dash:schema:mpd:2011 DASH-MPD.xsd"
  type="static"
  mediaPresentationDuration="PT3256S"
  minBufferTime="PT10.00S"
  profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">

  <BaseURL>http://cdn.example.com/movie23453235/</BaseURL>

  <Period>
    <!-- Audio protected with a specified license -->
    <AdaptationSet  mimeType="audio/mp4" codecs="mp4a.40" lang="en"
      subsegmentStartsWithSAP="1"
      subsegmentAlignment="true">
      <ContentProtection schemeIdUri="http://example.net/052011/drm">
        <drm:License>http://MoviesSP.example.com/protect?license=kljklsdfiowek</drm:License>
        <drm:Content>http://MoviesSP.example.com/protect?content=oyfYvpo8yFyvyo8f</drm:Content>
      </ContentProtection>
      <Representation id="1" bandwidth="64000">
        <BaseURL>audio/en/64.mp4</BaseURL>
      </Representation>
    </AdaptationSet>
    <!-- Audio protected with embedded information defined by 'ZZZZ' -->
    <AdaptationSet  mimeType="audio/mp4" codecs="mp4a.40" lang="fr"
      subsegmentStartsWithSAP="1"
      subsegmentAlignment="true">
      <ContentProtection schemeIdUri=" urn:mpeg:dash:mp4protection:2011" value="ZZZZ"/>
```

```
      <Representation id="3" bandwidth="64000">
        <BaseURL>audio/fr/64.mp4</BaseURL>
      </Representation>
    </AdaptationSet>
    <!-- Timed text in the clear -->
    <AdaptationSet  mimeType="application/ttml+xml" lang="de">
      <Representation id="5" bandwidth="256">
        <BaseURL>subtitles/de.xml</BaseURL>
      </Representation>
    </AdaptationSet>
    <!-- Video protected with a specified license -->
    <AdaptationSet  mimeType="video/mp4" codecs="avc1" subsegmentAlignment="true"
subsegmentStartsWithSAP="2">
      <ContentProtection schemeIdUri="http://example.net/052011/drm">
        <drm:License>http://MoviesSP.example.com/protect?license=jfjhwlsdkfiowkl</drm:License>
        <drm:Content>http://MoviesSP.example.com/protect?content=mslkfjsfiowelkfl</drm:Content>
      </ContentProtection>
      <BaseURL>video/</BaseURL>
      <Representation id="6" bandwidth="256000" width="320" height="240">
        <BaseURL>video256.mp4</BaseURL>
      </Representation>
      <Representation id="7" bandwidth="512000" width="320" height="240">
        <BaseURL>video512.mp4</BaseURL>
      </Representation>
      <Representation id="8" bandwidth="1024000" width="640" height="480">
        <BaseURL>video1024.mp4</BaseURL>
      </Representation>
    </AdaptationSet>
  </Period>
</MPD>
```

G.8 Example for usage of Role descriptor

In the following MPD example, "`supplementary`" audio Representations with ids "31" or "32" can be presented together with "main" video Representation with id "11" or "12" since **`Viewpoint`** descriptors are equivalent, i.e. the `@schemeIdUri` and the `@value` are equivalent. Similarly, the "`supplementary`" audio Representation with ids "41" or "42" can be presented together with "**`alternate`**" video Representations with ids "21" and "22".

NOTE The MPD is not complete and only provides a description of the concept.

```
<MPD>
    <Period>
        <AdaptationSet mimeType="video/mp4" group="1"/>
            <Role schemeIdUri="urn:mpeg:dash:role:2011" value="main"/>
            <Viewpoint schemeIdUri="urn:mpeg:dash:viewpoint:2011" value="vp1"/>
            <Representation id="11" bandwidth="1024000">…</Representation>
            <Representation id="12" bandwidth="512000">…</Representation>
                    …
        </AdaptationSet>

        <AdaptationSet mimeType="video/mp4" group="1"/>
            <Role schemeIdUri="urn:mpeg:dash:role:2011" value="main"/>
            <Viewpoint schemeIdUri="urn:mpeg:dash:viewpoint:2011" value="vp2"/>
            <Representation id="11" bandwidth="1024000">…</Representation>
            <Representation id="12" bandwidth="512000">…</Representation>
                    …
        </AdaptationSet>

        <AdaptationSet mimeType="audio/mp4" group="1"/>
            <Role schemeIdUri="urn:mpeg:dash:role:2011" value="main"/>
            <Role schemeIdUri="urn:mpeg:dash:role:2011" value="supplementary"/>
            <Viewpoint schemeIdUri="urn:mpeg:dash:viewpoint:2011" value="vp1"/>
            <Representation id="11" bandwidth="1024000">…</Representation>
            <Representation id="12" bandwidth="512000">…</Representation>
                    …
        </AdaptationSet>
```

```
        <AdaptationSet mimeType=”audio/mp4” group=”1”/>
            <Role schemeIdUri=”urn:mpeg:dash:role:2011” value=”main”/>
            <Role schemeIdUri=”urn:mpeg:dash:role:2011” value=”supplementary”/>
            <Viewpoint schemeIdUri=”urn:mpeg:dash:viewpoint:2011” value=”vp2”/>
            <Representation id=”11” bandwidth=”1024000”>…</Representation>
            <Representation id=”12” bandwidth=”512000”>…</Representation>
                    …
        </AdaptationSet>

    </Period>
    …
</MPD>
```

G.9 Example for usage of Event Messaging

In the following MPD example, two types of Events are added. In the Event stream with `@schemeIdUri="urn:org:example:xscte3"` time-synchronous events are added to the program. The events have a presentation time and a presentation duration. In addition, one Representation carries the MPD validity expiry information in an Inband event stream and an ad break information for ad insertion (assuming ANSI/SCTE 35 fields are used to represent the ad break parameters). The 'emsg' box may contain information as follows:

```
scheme_id_uri               = "urn:org:example:xscte35"
value                       = 0x0602
timescale                   = 1000
presentation_time_delta     = 8000
event_duration              = 0xFFFF
id                          = 12356789
message_data[] =
                    <metadata label="Ad Insertion Trigger"
                        type="http://www.example.com/schemas/xscte35"
                        namespace="xscte35"
                        xmlns:xscte35="http://www.example.com/schemas/xscte35">
                        <xscte35:signal>
                            <id name="segmentEventId" value="1" />
                            <time name="start" value="10" />
                            <time name="end" value="20" />
                        </xscte35:signal>
                    </metadata>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<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="PT30M"
    availabilityStartTime="2011-12-25T12:30:00"
    minBufferTime="PT4S"
    profiles="urn:mpeg:dash:profile:isoff-live:2011">

    <BaseURL>http://cdn1.example.com/</BaseURL>
    <BaseURL>http://cdn2.example.com/</BaseURL>

    <Period id="1">
        <EventStream schemeIdUri="urn:uuid:XYZY" timescale="1000" value="call">
            <Event presentationTime="0" duration="10000" id="0">+ 1 800 10101010</Event>
            <Event presentationTime="20000" duration="10000" id="1">+ 1 800 10101011</Event>
            <Event presentationTime="40000" duration="10000" id="2">+ 1 800 10101012</Event>
            <Event presentationTime="60000" duration="10000" id="3">+ 1 800 10101013</Event>
        </EventStream>
        <!-- Video -->
        <AdaptationSet
            mimeType="video/mp4"
```

```
            codecs="avc1.4D401F"
            frameRate="30000/1001"
            segmentAlignment="true"
            startWithSAP="1">
            <BaseURL>video/</BaseURL>
            <SegmentTemplate timescale="90000" initialization="$Bandwidth%/init.mp4v"
media="$Bandwidth%/$Time$.mp4v">
                <SegmentTimeline>
                    <S t="0" d="180180" r="432"/>
                </SegmentTimeline>
            </SegmentTemplate>
            <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>
        <!-- English Audio -->
        <AdaptationSet mimeType="audio/mp4" codecs="mp4a.40" lang="en" segmentAlignment="0"
startWithSAP="1">
            <SegmentTemplate timescale="48000" initialization="audio/en/init.mp4a"
media="audio/en/$Time$.mp4a">
                <SegmentTimeline>
                    <S t="0" d="96000" r="432"/>
                </SegmentTimeline>
            </SegmentTemplate>
            <Representation id="a0" bandwidth="64000">
                <InbandEventStream schemeIdUri="urn:mpeg:dash:event:2012"
value="1"></InbandEventStream>
                <InbandEventStream schemeIdUri="urn:org:example:event"
value="avail"></InbandEventStream>
            </Representation>
        </AdaptationSet>
        <!-- French Audio -->
        <AdaptationSet mimeType="audio/mp4" codecs="mp4a.40" lang="fr" segmentAlignment="0"
startWithSAP="1">
            <SegmentTemplate timescale="48000" initialization="audio/fr/init.mp4a"
media="audio/fr/$Time$.mp4a">
                <SegmentTimeline>
                    <S t="0" d="96000" r="432"/>
                </SegmentTimeline>
            </SegmentTemplate>
            <Representation id="a0" bandwidth="64000" />
        </AdaptationSet>
    </Period>
</MPD>
```

Bibliography

[1] ISO/IEC 23001-7, *Information technology — MPEG systems technologies — Part 7: Common encryption in ISO base media file format files*

[2] 3GPP TS 26.234, *Transparent end-to-end packet switched streaming service (PSS); Protocols and codecs,* http://www.3gpp.org/ftp/Specs/html-info/26234.htm

[3] 3GPP TS 26.244, *Transparent end-to-end packet switched streaming service (PSS); 3GPP file format (3GP)*, http://www.3gpp.org/ftp/Specs/html-info/26244.htm

[4] IETF RFC 1952, *GZIP file format specification version 4.3*, May 1996

[5] IETF RFC 2231, *MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations*, November 1997

[6] IETF RFC 2405, *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*, November 1996

[7] IETF RFC 2818, *HTTP Over TLS*, May 2000

[8] IETF RFC 5905, *Network Time Protocol Version 4: Protocol and Algorithms Specification*, June 2010

[9] ITU-T Rec.H.262 | ISO/IEC 13818-2, *Information technology — Generic coding of moving pictures and associated audio information: Video*

[10] “XML Encryption Syntax and Processing Version 1.1”, W3C Recommendation 11 April 2013, http://www.w3.org/TR/xmlenc-core/

[11] “XML Signature Syntax and Processing (Second Edition)”, W3C Recommendation 10 June 2008, http://www.w3.org/TR/xmldsig-core/

[12] IEEE 1003.1-2008, *IEEE Standard for Information Technology – Portable Operating System Interface (POSIX)*, Base Specifications, Issue 7.

ICS 35.040

Price based on 144 pages