ETSI TS 126 247 V10.0.0 (2011-06)

Technical Specification

Universal Mobile Telecommunications System (UMTS);
LTE;
Transparent end-to-end Packet-switched Streaming Service (PSS);
Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH)
(3GPP TS 26.247 version 10.0.0 Release 10)

Reference
RTS/TSGS-0426247va00

Keywords
LTE, UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp.

Contents

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

The 3GPP transparent end-to-end packet-switched streaming service (PSS) specification consists of seven 3GPP TSs: 3GPP TS 22.233 [1], 3GPP TS 26.233 [2], 3GPP TS 26.234 [3], 3GPP TS 26.244 [4], 3GPP TS 26.245 [5], 3GPP TS 26.246 [6], and the present document.

The TS 22.233 contains the service requirements for the PSS. The TS 26.233 provides an overview of the PSS. The TS 26.234 provides the details of the protocols and codecs used by the PSS. The TS 26.244 defines the 3GPP file format (3GP) used by the PSS and MMS services. The TS 26.245 defines the Timed text format used by the PSS and MMS services. The TS 26.246 defines the 3GPP SMIL language profile. The present document defines Progressive Download and Dynamic Adaptive Streaming over HTTP.

The TS 26.244, TS 26.245 and TS 26.246 start with Release 6. Earlier releases of the 3GPP file format, the Timed text format and the 3GPP SMIL language profile can be found in TS 26.234.

The TS 26.247 starts with Release 10. Earlier releases of Progressive Download and Dynamic Adaptive Streaming over HTTP can be found in TS 26.234.

Introduction

Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH) collects a set of technologies how progressive download and adaptive streaming of continuous media may be carried out exclusively over HTTP.

1 Scope

The present document specifies Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH). This specification is part of Packet-switched Streaming Service (PSS). HTTP-based progressive download and dynamic adaptive streaming are separated from TS 26.234 to differentiate from RTP-based streaming that is maintained in TS 26.234. HTTP-based progressive download and dynamic adaptive streaming may be deployed independently from RTP-based PSS, for example by using standard HTTP/1.1 servers for hosting data formatted as defined in the present document.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document.*

[1] 3GPP TS 22.233: "Transparent End-to-End Packet-switched Streaming Service; Stage 1".

[2] 3GPP TS 26.233: "Transparent end-to-end Packet-switched Streaming service (PSS); General description".

[3] 3GPP TS 26.234: "Transparent end-to-end packet switched streaming service (PSS); Protocols and codecs".

[4] 3GPP TS 26.244: "Transparent end-to-end packet switched streaming service (PSS); 3GPP file format (3GP)".

[5] 3GPP TS 26.245: "Transparent end-to-end packet switched streaming service (PSS); Timed text format".

[6] 3GPP TS 26.246: "Transparent end-to-end packet switched streaming service (PSS); 3GPP SMIL Language Profile".

[7] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[8] IETF STD 0007: "Transmission Control Protocol", Postel J., September 1981.

[9] IETF RFC 2616: "Hypertext Transfer Protocol – HTTP/1.1", Fielding R. et al., June 1999.

[10] Open Mobile Alliance, Service and Content Protection for Mobile Broadcast Services, Approved Version 1.0, February 2009.

[11] ISO/IEC 14496-12:2005 | 15444-12:2005: "Information technology – Coding of audio-visual objects – Part 12: ISO base media file format" | "Information technology – JPEG 2000 image coding system – Part 12: ISO base media file format".

[12] IETF RFC 2818: "HTTP Over TLS", E. Rescorla, May 2000.

[13] IETF RFC 5646: "Tags for Identifying Languages", A. Phillips, M. Davis, September 2009.

[14] IETF RFC 4281: "The Codecs Parameter for Bucket Media Types", R. Gellens, D. Singer, P. Frojdh, November 2005.

[15] Open Mobile Alliance: "DRM Content Format V 2.0".

[16] Open Mobile Alliance: "DRM Content Format V 2.1".

[17] IETF RFC 3986: "Uniform Resource Identifiers (URI): Generic Syntax", Berners-Lee T., Fielding R. and Masinter L., January 2005.

[18] IETF RFC 1952: "GZIP file format specification" version 4.3, P. Deutsch, May 1996.

[19] IETF RFC 1738: "Uniform Resource Locators (URL)", December 1994.

[20] W3C XLINK: "XML Linking Language (XLink)" Version 1.1, W3C Recommendation 06, May 2010.

[21] IETF RFC 3406: "Uniform Resource Names (URN) Namespace Definition Mechanisms", October 2002.

[22] OMA-ERELD-DM-V1_2-20070209-A: "Enabler Release Definition for OMA Device +Management, Approved Version 1.2"

[23] 3GPP TS 33.310: "Network Domain Security (NDS); Authentication Framework (AF)".

[24] IETF RFC 2045: "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies".

[25] IETF RFC 2231: " MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations".

3 Definitions, abbreviations and conventions

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TR 21.905 [7] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in TR 21.905 [7].

available segment: a Segment which is accessible at its assigned HTTP-URL, possibly restricted by a byte range, i.e. the request with an HTTP GET results in a reply of the Segment and a 2xx OK status code.

continuous media: media with an inherent notion of time. In the present document speech, audio, video, timed text and timed graphics.

HTTP-URL: a URI with a fixed scheme of 'http://' or https://.

media component: A media component is an encoded version of one individual media type such as audio, video or timed text.

media content: A set of media components, (e.g. audio, video, timed text) that have a common timeline as well as relationships one how they may be presented for example individually, jointly, or mutually exclusive. An example for a media content is a program or a movie.

media presentation: a structured collection of data that is accessible to the 3GP-DASH client.

media presentation description (MPD): contains information describing the media presentation and that is required by the 3GP-DASH client to construct appropriate information and to provide the streaming service to the user.

media time: a continuous timeline inherent to the media that describes decoding and presentation timestamps of media samples.

period: A period is a timely subset of the media presentation. The sequence of periods constitutes the media presentation. Periods are consecutive and non-overlapping.

representation: A structured collection of data which contains one or more media components with specific attributes, e.g. bandwidth, language, resolution etc.

representation access point (RAP): position in a media segment that is identified as being a position for which it is possible to start playback using only the information contained in the media segment from that position onwards (preceded by initialising with the initialisation segment, if any). It consists of a byte index, I_{RAP}, and a presentation time, T_{RAP}, related as follows

- T_{RAP} is the earliest presentation time such that all access units with presentation time greater than or equal to T_{RAP} can be correctly decoded using stream data starting at I_{RAP} and no stream data before I_{RAP}.
- I_{RAP} is the greatest byte index in the stream such that all access units with presentation time greater than or equal to T_{RAP} can be correctly decoded using stream data starting at I_{RAP} and no stream data before I_{RAP}.

Representation access points may coincide with random access points in certain media streams..

segment: a unit of media that can be referenced by an HTTP-URL, possibly restricted by a byte range, included in the MPD.

segment availability end time: the time instant in wall-clock time at which a Segment ceases to be an available Segment.

segment availability start time: the time instant in wall-clock time at which a Segment becomes an available Segment.

valid segment URL: an HTTP-URL that is promised to reference a segment during its segment availability period.

wall-clock time: time as stated by UTC (Universal Co-ordinated Time).

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in TR 21.905 [7] and the following apply.
An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in TR 21.905 [7].

3GP	3GPP file format
3GP-DASH	3GPP Dynamic Adaptive Streaming over HTTP
AHS	Adaptive HTTP Streaming
AVC	Advanced Video Coding
DM	Device Management
DRM	Digital Rights Management
HSD	HTTP Streaming and Download
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
MPD	Media Presentation Description
MPEG-2 TS	Moving Picture Experts Group Transport Stream
MIME	Multipurpose Internet Mail Extensions
OMA	Open Mobile Alliance
PDCF	Packetized DRM Content Format
PSS	Packet-switched Streaming Service
QoE	Quality-of-Experience
RAP	Representation Access Point
RFC	Request For Comments
RTP	Real-time Transport Protocol
SMIL	Synchronised Multimedia Integration Language
TLS	Transport Layer Security
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UTC	Universal Time Coordinated
UTF-8	Unicode Transformation Format (the 8-bit form)
W3C	WWW Consortium
XML	eXtensible Markup Language
XSLT	eXtensible Stylesheet Language Transformation

3.3 Conventions

The following naming conventions apply in this specification:

- 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 is constructed of two or more combined words, camel-casing is typically used, e.g. **`ImportantElement`**.

- 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`**, we may write **`Element`**`@attribute`. If an attribute is constructed of two or more combined words, camel-casing is typically used after the first word, e.g. `@veryImportantAttribute`.

- Variables defined in the context of the present 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. Media Presentation, Period, Group, Adaptation Set, Representation, Segment, etc.

4 Overview

The present document specifies Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH) for continuous media. The features are separated from the umbrella specification TS 26.234 [3] to differentiate from RTP-based streaming that is specified and maintained in TS 26.234. Services relying exclusively on these features may be deployed independently from RTP-based PSS servers, for example by using standard HTTP/1.1 servers for hosting the services.

The specification covers the following aspects:

- System Description: describes the relationship to the PSS architecture and refines the architecture, interfaces and protocols that are defined in this specification.

- Progressive Download over HTTP.

- 3GPP Dynamic Adaptive Streaming over HTTP (3G-DASH) provides an overview of the architecture, the formats and the models that build the basis for 3GP-DASH. Also, 3GP-DASH Profiles provides an identifier and refers to a set of specific restrictions in this or other specifications.

- DASH - Media Presentation describes the data model of a Media Presentation. It also provides an overview on elements and attributes that may be used to describe components and properties of a media presentation in a Media Presentation Description (MPD).

- DASH - Usage of the 3GP file format defines how segments can be formed based on the 3GP file format.

- Quality-of-Experience for Progressive Download and 3GP-DASH.

- Normative annexes for MPD schema (Annex B), Descriptor Scheme Definitions (Annex C), OMA DM QoE Management Object (Annex F), File format extensions for 3GPP DASH support (Annex G) and MIME Type Registration for MPD (Annex H).

- Informative annexes for Client Behaviour (Annex A), MPD Examples (Annex D), and Mapping MPD structure and semantics to SMIL (Annex E).

5 System Description

5.1 Overview

Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH) enables to provide services to deliver continuous media content over Hypertext Transfer Protocol (HTTP) in a sense that all resources that compose the service are accessible through HTTP-URLs and the HTTP/1.1 protocol as specified in RFC 2616 [9] may be used to deliver the metadata and media data composing the service. This enables that standard HTTP servers and standard HTTP caches can be used for hosting and distributing continuous media content. Figure 1 shows the architecture for services using progressive download and Figure 2 shows the architecture for services using 3GP-DASH.

The present document deals with the specification of interfaces between the Client and the Server. Specifically, it defines the formats that may be delivered exclusively over the HTTP interface to enable progressive download and streaming
services.

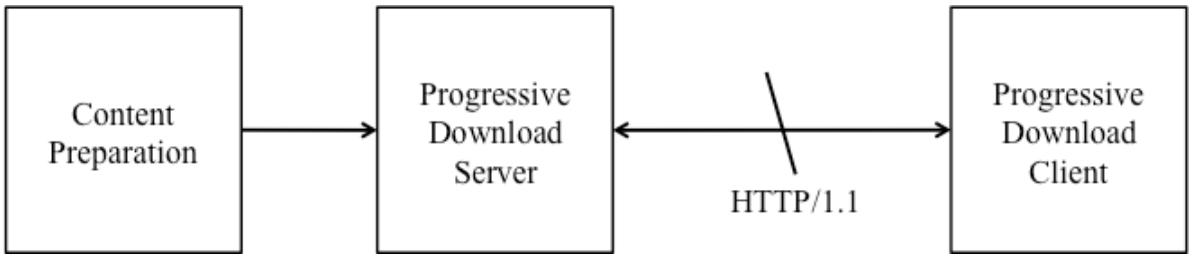

Figure 1: Architecture for Progressive Download over HTTP

Services using the features described in this specification may be deployed within PSS as specified in TS 26.233 [2] and TS 26.234 [3]. In this case the Progressive Download/3GP-DASH Server may be a sub-function of the PSS server and the Progressive Download/3GP-DASH client may be a sub-function of the PSS client.

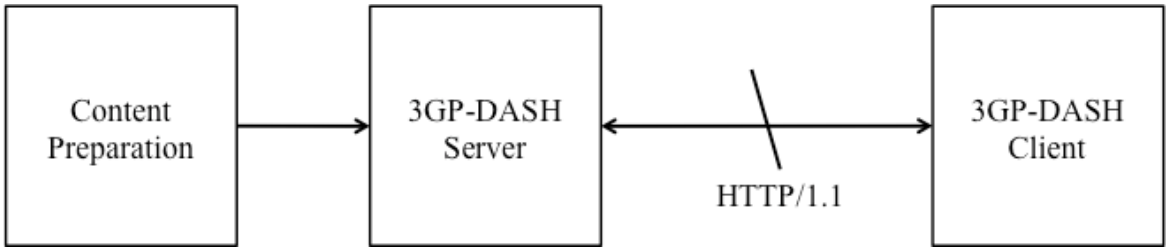

Figure 2: Architecture for 3GP-DASH

Services using the features defined in this specification may also be deployed independent of the PSS servers and clients. In this case the Progressive Download/3GP-DASH client shall support the formats and codecs according to this specification.

Access to services based on the features defined in the present document is introduced in clause 5.2.

The protocol support for services using the features defined in this specification is provided in clause 5.3.

Clients supporting progressive download-based services shall support the features and formats as specified in clause 6 of this specification.

Clients supporting 3GP-DASH shall support the features and formats as specified in clause 7 of this specification.

Clients supporting QoE Metrics and Reporting shall support the features as specified in clause 10 of this specification.

5.2 Service Access

Service access refers to the method by which a Client initially accesses the service. Service access for services based in the specification can be achieved e.g. by a Media Presentation Description or a URL to the media file.

The service access URL can be made available to a client in many different ways. Clients supporting services based on the features in this specification shall be able to access services that are provided through an HTTP-URL. However, it is out of the scope of this specification to mandate any specific mechanism. A preferred way may be to embed URLs for service establishment within HTML pages.

5.3 Protocols

Progressive Download and 3GP-DASH clients shall comply with a *client* as specified in RFC 2616 [9]. The resource hosting the 3GP files and DASH Segments shall comply with a *server* as specified in RFC 2616 [9].

Progressive Download and 3GP-DASH clients should use the HTTP GET method or the HTTP partial GET method, as specified in RFC 2616 [9], clause 9.3, to access media offered at HTTP-URLs.

Figure 3 shows a protocol stack for services in the context of this specification. 3GP Files in progressive download as well as Segments based on the 3GPP File Format shall be accessible through HTTP.

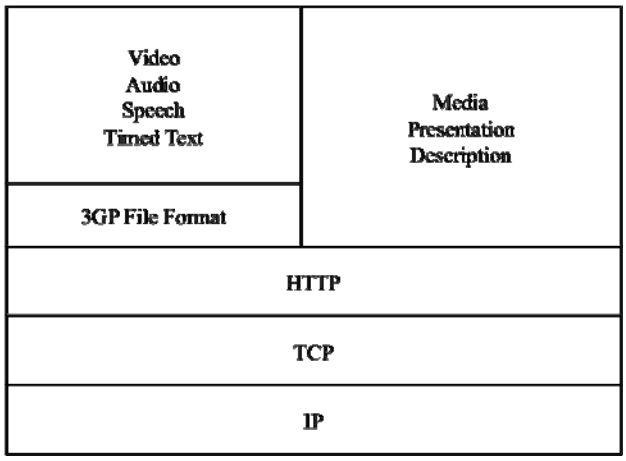

Figure 3: Overview of the protocols stack

Transport security in Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH) is achieved using the HTTPS (Hypertext Transfer Protocol Secure) specified in RFC 2818 [12] and TLS as specified in TLS profile of Annex E in TS 33.310 [23]. In case secure delivery is desired, HTTPS should be used to authenticate the server and to ensure secure transport of the content from server to client.

NOTE: The use of HTTPS for delivering Media Segments may inhibit caching at proxies and add overhead at the server and the client.

6 Progressive Download over HTTP

6.1 General

As an alternative to conventional streaming, a client may download, typically through HTTP, a media file that encapsulates continuous media and may play the media from the local storage. A PSS client shall support progressive download and playout of 3GP files [4] as specified in the remainder of this clause.

The media file encapsulating the continuous media is accessed directly by issuing one or more HTTP GET or partial GET requests to the referenced media file. An example of a valid URL is http://example.com/morning_news.3gp.

6.2 Progressive Download

Progressive download uses normal HTTP download using HTTP GET or partial GET requests. The differences between regular download and Progressive Download are that 1) the content may be authored as progressively downloadable, and 2) the terminal recognises that the content is suitable for progressive download.

A client downloading continuous media may decide to start playout of the encapsulated media data before the download of the media file is completed.

6.3 3GPP File Format Profiles

The following profiles of the 3GPP file format in TS 26.244 [4] shall be supported by clients supporting Progressive Download over HTTP:

- Basic profile, and
- Progressive-download profile.

7 3GPP Dynamic Adaptive Streaming over HTTP

7.1 System Description

The 3GPP Dynamic Adaptive Streaming over HTTP (3GP-DASH) specified in this specification provides streaming services over HTTP. This enables delivering content from standard HTTP servers to an HTTP-Streaming client and enables caching content by standard HTTP caches.

The specification for 3GP-DASH primarily defines two formats:

1) 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. In the context of this part of the standard, the resource identifiers are exclusively HTTP-URLs,. However, this specification additionally enables the restriction of these URLs by a byte range attribute.

2) The Segment formats specify the formats of the entity body of the request response when issuing a HTTP GET request or a partial HTTP GET with the indicated byte range through HTTP/1.1 as defined in RFC 2616 [9] to a resource identified in the MPD.

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 specification is exclusively HTTP/1.1. Such a client is referred to as a 3GP-DASH client in the remainder of the present document. However, this specification does not provide a normative definition for such a client. An informative client model to illustrate the formats defined in this specification is provided in section 2. An informative example client behaviour description is provided in Annex A of this specification.

Figure 4 shows an architecture in which the formats defined in this specification are typically 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 specification deals with the definition of formats that are accessible on the interface to the 3GP-DASH client, indicated by the solid lines. Any other formats or interfaces are not in scope of this specification. In the considered deployment scenario, it is assumed that the 3GP-DASH client has access to an MPD. The MPD provides sufficient information for the 3GP-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 data appropriately.

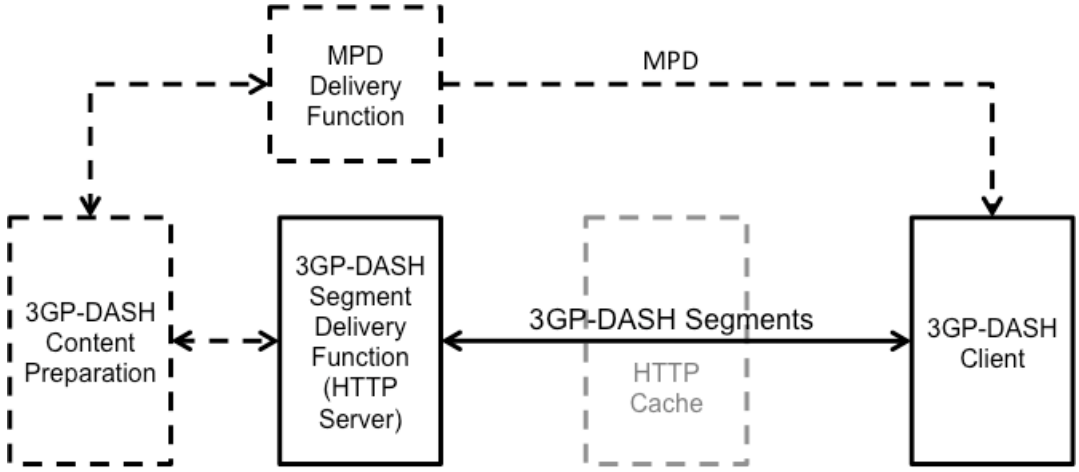

Figure 4: System Architecture for 3GP-DASH

The normative aspects of 3GP-DASH formats are defined by

- the profiles defined in clause 7.3.
- the DASH Media Presentation as defined in clause 8.
- the usage of the 3GPP file format for DASH as defined in clause 9.

The clauses mentioned above may refer to normative aspects in clause 10 on Quality-of-Experience as well as to normative Annexes B, C, E, G, and H.

7.2 3GP-DASH Client Model

The design of the formats defined in this standard is based on the informative client model as shown in Figure 5. The figure illustrates the logical components of a conceptual 3GP-DASH client model. In this figure the 3GP-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 standard, the output of the DASH Access Engine consists of media in container formats according to the ISO/IEC 14496-12 ISO Base Media File Format [11] and specifically the 3GP file format [4]. In addition, timing information is provided that maps the internal timing of the media to the time line of the Media Presentation.

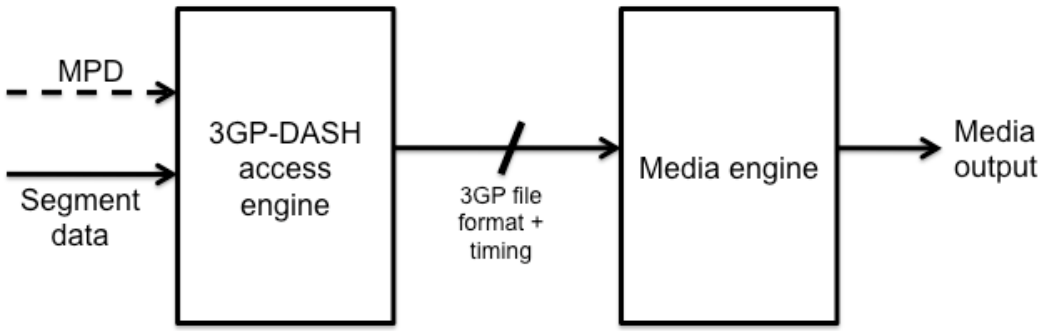

Figure 5: 3GP-DASH client Model

7.3 3GP-DASH Profiles

7.3.1 General

Profiles of 3GP-DASH are defined so as to enable interoperability and the signaling of the use of features etc.

A profile has an identifier and refers to a set of specific restrictions. Those restrictions might be on features of the media presentation description (MPD) as defined in clause 8 of this specification, segment formats as for example defined in clause 9 of this specification, usage of the network, codec(s) used, content protection formats, or on quantitative measures such as bit-rates, segment lengths, screen size, and so on. Profiles defined in this specification define restrictions on features of this specification, but may additionally impose restrictions on other aspects of media delivery.

A profile is a claim and a permission; it claims that the media presentation (MPD document and segment formats) conforms to the profile, and gives permission to a client that implements that profile to read the media presentation, interpret what it recognizes, and ignore the material it does not understand.

The profiles with which a Media Presentation complies are indicated in the **MPD**@profiles attribute. This element is a space-delimited list of profile identifiers each of which is a URI. Profile identifiers defined in this specification are URNs conforming to RFC 3406 [21]. URLs may also be used. 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.

7.3.2 3GPP Adaptive HTTP Streaming (Release-9 AHS)

The Release-9 AHS profile is identified by the URN 'urn:3GPP:PSS:profile:AHS9'. This includes all features defined TS 26.234 [3] Release-9, clause 12.

7.3.3 3GP-DASH Release-10 Profile

7.3.3.1 Introduction

The 3GP-DASH Release-10 profile is identified by the URN 'urn:3GPP:PSS:profile:DASH10'. This includes all features defined in the Release-10 version of this specification in clauses 7.3.3.2, 7.3.3.3, 8, 9 and 10.

7.3.3.2 Media Codecs

For the 3GP-DASH Release-10 profile clients supporting a particular continuous media type, the corresponding media decoders are specified in TS 26.234 [3], clause 7.2 for speech, 7.3 for audio, 7.4 for video, 7.9 for timed text and 7.11 for timed graphics.

7.3.3.3 Content Protection

For the 3GP-DASH Release-10 profile clients content protection may support OMA DRM 2.0 [15] or OMA DRM 2.1 [16]. Other content protection schemes may be supported. The ContentProtection element in the MPD should be used to convey content protection information.

When using OMA DRM V2.0 or OMA DRM V2.1 scheme for content protection, the non-streamable Packetized DRM Content Format (PDCF) shall be used. An OMA-DRM encrypted Representation shall include the brands '3gh9' and 'opf2'. OMA-DRM [15] [16] defines the procedures for acquiring the Rights Object from the Rights Issuer to decrypt PDCF protected content. The scheme is identified by a **ContentProtection**@schemeIdUri set to "urn:mpeg:mpegB:dash:mp4protection:odkm"

8 DASH - Media Presentation

8.1 Introduction

A Media Presentation is a structured collection of data that is accessible to a 3GP-DASH client to provide a streaming service to the user.

A Media Presentation is described in a Media Presentation Description (MPD) including any possible updates of the MPD. The MPD is defined in clause 8.2 and the update mechanisms in 8.5. Assembly of a fragmented MPD is defined in 8.3. The data model that constitutes a Media Presentation is defined in 8.4 and some additional elements in the MPD that describe the content are provided in 8.6.

8.2 Media Presentation Description

8.2.1 General

The Media Presentation Description (MPD) is a document that contains metadata required by a 3GP-DASH client to construct appropriate HTTP-URLs to access Segments and to provide the streaming service to the user. HTTP-URLs may be absolute or relative. If relative then reference resolution as defined in 8.2.3 shall be applied. Handling of alternative base URLs is addressed in 8.2.4.

The MPD is an XML-document that is formatted according to the XML schema provided in clause 8.2.2.

The MPD shall be authored such that, after unrecognized XML attributes or elements are removed, the result is a valid XML document formatted according to the XML schema provided in clause 8.2.2 and that complies with this standard.

The MIME type of the MPD shall be '`application/dash+xml`' as defined in Annex H.1.

MPDs may be updated as specified in clause 8.5. Updates may also be done using MPD delta files as defined in clause 8.5.2. The MIME type of an MPD delta file shall be '`application/dashdelta+xml`' as defined in Annex H.2.

The delivery of the MPD is not in scope of this specification. If the MPD is delivered over HTTP, then the MPD may be content encoded for transport, as described in [18] using the generic GZip algorithm RFC 1952 [18]. 3GP-DASH clients shall support GZip content decoding of the MPD when delivered over HTTP (GZIP RFC 1952 [18], clause 9).

An adaptive HTTP streaming client shall ignore any XML attributes or elements in a valid XML document formatted according to the XML schema provided in clause 8.2.2 that it does not recognize. If attributes or elements not defined in the schema in clause 8.2.2 are added to the MPD in the same namespace, the MPD shall be authored such that DASH client gets a valid and functional MPD.

8.2.2 Schema

The XML schema of the MPD is provided below. Specific types, elements and attributes are introduced in the remainder of this clause. The complete MPD schema is provided in Annex B of this specification. In case of any inconsistencies the schema in Annex B takes precedence over the XML-syntax snippets provided in this clause. For the normative schema refer to the schema in Annex B.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009"
    attributeFormDefault="unqualified" elementFormDefault="qualified"
    xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
    <xs:annotation>
        <xs:appinfo>Media Presentation Description</xs:appinfo>
        <xs:documentation xml:lang="en"> This Schema defines 3GPP Media Presentation Description!
        </xs:documentation>
    </xs:annotation>

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

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

</xs:schema>
```

8.2.3 Reference Resolution

URLs at each level of the MPD are resolved 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 [17], Clause 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 RFC 3986 then the MPD should not be interpreted.

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

- On MPD level in **MPD.BaseURL** element, see 8.2.1.
- On Period level in **Period.SegmentInfoDefault.BaseURL** element (for details refer to 8.4.2). The level above the period level is the MPD level.
- On Adaptation Set level in **AdaptationSet.SegmentInfoDefault.BaseURL** element (for details refer to 8.4.3.3). The level above the group level is the Period level.
- On Representation level in **SegmentInfo.BaseURL** (for details refer to 8.4.3.4). The level above the Representation level is the Adaptation Set level.

8.2.4 Alternative Base URLs

If alternative base URLs are provided through the **BaseURL** element at any level, this means that the identical segments are accessible at multiple locations. In the absence of other criteria, the 3GP-DASH client may use the first base URL as 'base URI'. The 3GP-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.

8.3 MPD Assembly

8.3.1 Introduction

This clause defines a mechanism for referencing a remote DASH element from within a local MPD. A subset of W3C XLINK [20] simple links is defined consisting of:

- restricted syntax and semantics in clause8.3.2, and
- the processing model in clause 8.3.3.

8.3.2 Syntax and semantics

Table 1 provides the XLINK attributes that are used in this standard.

Table 1: XLINK attributes used in this specification

Attribute	Comments and Usage
`@xlink:type`	Identifies the type of W3C XLINK being used. In the context of standard, all references shall be W3C XLINK simple links. As the attribute `@xlink:type` is optional with fixed setting `@xlink:type="simple"`.
`@xlink:href`	Identifies the remote DASH Element by URI as defines in IETF RFC 3986 [17]. In the context of this standard, URI shall exclusively be HTTP-URLs.
`@xlink:show`	Defines the desired behaviour of a remote DASH element once dereferenced from within a MPD as defined in W3C XLINK. In the context of this specification 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 DASH element, is not defined. Thus, the actual behaviour for this standard is defined in clause 8.3.3.

Attribute	Comments and Usage
`@xlink:actuate`	Defines the desired timing of dereferencing a remote DASH-Element from within a MPD as defined in W3C XLINK. The following attribute values are allowed in this standard: 1) `onLoad`: an application should dereference the remote DASH element immediately on loading the MPD. 2) `onRequest` (default): formally, an application should dereference the remote DASH-element only on a post-loading event triggered for the purpose of dereferencing. In the context of this specification, the application dereferences the link only for those resources it needs (or anticipates it probably will need). Examples include de-referencing a link in a **`Period`** element when the play-time is expected to enter that period, de-referencing a representation group link when it appears to contain representations that will be needed, and so on.

The restricted schema for XLINK in the context of the standard is referred to as "xlink.xsd" in any schema in this standard and defined is as follows:

```
<?xml version='1.0' encoding='UTF-8'?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://www.w3.org/1999/xlink"
    xmlns:xlink="http://www.w3.org/1999/xlink">

    <xs:attribute name="type" type="xs:token" fixed="simple"/>

    <xs:attribute name="href" type="xlink:hrefType"/>

    <xs:simpleType name="hrefType">
        <xs:restriction base="xs:anyURI"/>
    </xs:simpleType>

    <xs:attribute name="show" type="xs:token" fixed="embed"/>

    <xs:attribute name="actuate" type="xlink:actuateType" default="onRequest"/>

    <xs:simpleType name="actuateType">
        <xs:restriction base="xs:token">
            <xs:enumeration value="onLoad"/>
            <xs:enumeration value="onRequest"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>
```

8.3.3 Processing

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

1) URI references to remote DASH elements that cannot be resolved shall be treated as invalid references.

2) URI references to remote DASH elements that are inappropriate targets for the given reference shall be treated as invalid references (see list below for the appropriate targets).

3) URI references that directly or indirectly reference themselves are treated as invalid circular references.

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

1) Attributes shall be added to the element of the MPD that contains `@xlink:href` merged with existing attributes. If the same attributes are present in both MPD and remote DASH element, the attribute values should be the same. If they are not identical, then the value of the attribute of the MPD takes precedence over the value of the attribute in the remote DASH element.

2) The remote DASH element referenced by the `@xlink:href` shall conform to the type definition of the element in the MPD that contains `@xlink:href`.

3) All XLINK attributes shall be removed after dereferencing is completed.

3) Only a single element shall be included in a remote DASH element.

8.4 Hierarchical Data Model

8.4.1 General

A Media Presentation is described in the **MPD** element that is contained in an MPD document formatted as defined in clause 8.2.

A Media Presentation consists of:

- A sequence of one or more Periods described in 8.4.2.
- Each Period contains one or more Groups described in 8.4.3.1.
- Each Group contains one or more Adaptation Sets as defined in 8.4.3.3.
- Each Adaptation Sets contains one or more Representations as described in 8.4.3.4.
- Each Representation consists of one or more Segments. Segment Information is introduced in 8.4.4. Segments contain media data and/or metadata to access, decode and present the included media content.

This **MPD** element provides descriptive information that enables a client to choose Representations. For doing so, it provides descriptions of the Representations that may also be deduced by inspecting this Representation if available partially or in its entirety to the client. However, actual playback of the Representations is not controlled by the MPD information. Playback is controlled by the media engine operating on the media data in the usual way.

The Media Presentation timeline is defined by the concatenation of the timeline of each Period.

NOTE The playout procedure of the media may need to be adjusted at the end of the preceding Period to match the start time of the new Period as there may be small overlaps or gaps with a Representation at the end of the preceding Period.

The timeline in each Period is common to all Representations.

The summary of the semantics of the attributes and elements within an **MPD** element are provided in Table 2. The XML-syntax of the **MPD** element is provided in Table 3.

Table 2: Semantics of `MPD` element

Element or Attribute Name	Use	Description
MPD	1	The root element that carries the Media Presentation Description for a Media Presentation.
`@profile`	O	specifies a space delimited list specifying the Media Presentation profiles.
`@type`	OD default: OnDemand	'`OnDemand`' or '`Live`'. specifies the type of the Media Presentation. Currently, on-demand and live types are defined.
`@availabilityStartTime`	CM Must be present for type='Live'	For `@type='Live'` this attribute shall be present. In this case it specifies the anchor for the computation of the segment availability start time for any Segment in the Media Presentation. For `@type='OnDemand'` , 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 be available.
`@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	If present, it specifies the duration of the entire Media Presentation. In this case the attribute **MPD**`@minimumUpdatePeriodMPD` shall not be present.

Element or Attribute Name	Use	Description
		If the attribute is not present, the duration of the Media Presentation is unknown. In this case the attribute **MPD**@minimumUpdatePeriodMPD shall be present.
@minimumUpdatePeriodMPD	O	If this attribute is present, updates to the MPD are expected. The use of the value of this attribute is specified in clause 8.5. In this case the attribute **MPD**@mediaPresentationDuration shall not be present. If not present the minimum update period is assumed to be infinite and the attribute **MPD**@mediaPresentationDuration shall be present.
@minBufferTime	O	specifies the minimum amount of initially buffered media that is needed to ensure smooth playout provided that each Representation is continuously delivered at or above the value of its @bandwidth attribute.
@timeShiftBufferDepth	O	specifies the duration of the time shifting buffer that is guaranteed to be available for a Media Presentation with type 'Live'. When not present, the value is unknown. This value of the attribute is undefined if the @type attribute is equal to "OnDemand".
@suggestedPresentationDelay	O	specifies a fixed delay offset in time from the segment availability start times of each media segment that is suggested to be used by clients to enable synchronouous presentation of the media presentation with clients that also use this attribute. This value of the attribute is undefined if the @type attribute is equal to "OnDemand"
ProgramInformation	0…1	specifies descriptive information about the program. For more details refer to the description in clause 8.6.2.
DeltaSupport	0…N	If present, this element specifies that MPD delta files are supported by the server. For more details refer to the description in clause 8.5.2.
Location	0…N	specifies an absolute URL where the MPD is available.
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 clauses 8.2.3 and 8.2.4.
QualityMetrics	0..1	specifies information about the requested QoE reporting. For more details refer to clause 10.3. At most one **QualityMetrics** element shall be present in the MPD. NOTE: The schema allows more than one **QualityMetrics** elements for potential future extensions.
Period	1…N	specifies a Period. For more details refer to the description in clause 8.4.2.
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 3: Syntax of MPD element

```
    <!-- MPD Type -->
    <xs:complexType name="MPDtype">
        <xs:sequence>
            <xs:element name="ProgramInformation" type="ProgramInformationType" minOccurs="0"/>
            <xs:element name="DeltaSupport" type="DeltaSupportType" minOccurs="0"/>
            <xs:element name="Location" type="MpdUrlType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="BaseURL" type="BaseUrlType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="QualityMetrics" type="QualityMetricsType" minOccurs="0"
                maxOccurs="unbounded"/>
            <xs:element name="Period" type="PeriodType" maxOccurs="unbounded"/>
            <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="profiles" type="UriVectorType"/>
        <xs:attribute name="type" type="PresentationType" default="OnDemand"/>
        <xs:attribute name="availabilityStartTime" type="xs:dateTime"/>
```

```
        <xs:attribute name="availabilityEndTime" type="xs:dateTime"/>
        <xs:attribute name="mediaPresentationDuration" type="xs:duration"/>
        <xs:attribute name="minimumUpdatePeriodMPD" type="xs:duration"/>
        <xs:attribute name="minBufferTime" type="xs:duration"/>
        <xs:attribute name="timeShiftBufferDepth" type="xs:duration"/>
        <xs:attribute name="suggestedPresentationDelay" type="xs:duration"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- Type for space delimited list of URIs -->
    <xs:simpleType name="UriVectorType">
        <xs:list itemType="xs:anyURI"/>
    </xs:simpleType>

    <!-- Type of presentation - live or on-demand -->
    <xs:simpleType name="PresentationType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="OnDemand"/>
            <xs:enumeration value="Live"/>
        </xs:restriction>
    </xs:simpleType>

    <!-- Supplementary URL to the one given as attribute -->
    <xs:complexType name="BaseUrlType">
        <xs:simpleContent>
            <xs:extension base="xs:anyURI">
                <xs:anyAttribute namespace="##other" processContents="lax"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

    <xs:complexType name="MpdUrlType">
        <xs:simpleContent>
            <xs:extension base="xs:anyURI">
                <xs:anyAttribute namespace="##other" processContents="lax"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
```

8.4.2 Period

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

The attributes and elements contained in the **`Period`** element are provided in Table 4 along with their semantics. The XML syntax or the **`Period`** element is provided in Table 5.

For live services, the `@start` attribute of the first Period shall be present unless the first Period is an Early Available Period (see below for more details). For on-demand services the `@start` attribute of the first Period, if present, shall be zero. If not present a default value of zero is assumed.

Each Period has a conceptual start time *PeriodStart* in the Media Presentation that is relative to the start of the first Period. Period elements shall be physically ordered in the **`MPD`** element in increasing order of their *PeriodStart* time.

The *PeriodStart* time is determined as follows:

- If the attribute `@start` is present in the **`Period`**, then *PeriodStart* is identical to the value of this attribute and it overrides any possibly present prior `@duration` attribute information.
- If the `@start` attribute is absent, the previous **`Period`** element shall contain the `@duration` attribute unless the Period is an Early Available Period (see below for more details). Then the start time of the new Period *PeriodStart* is determined as the sum of the start time of the previous Period *PeriodStart* and the value of the attribute `@duration` of the previous Period.

Each Period extends until the *PeriodStart* of the next Period, or until the end of the Media Presentation in the case of the last Period.

PeriodStart times reflect the actual time that should elapse after playing the media of all prior Periods in this Media Presentation.

An Early Available Period is documented by a **Period** element for which the the **Period**@start attribute is not present and the **Period**@duration attribute of previous **Period** is also absent. 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 gets known through an update of the MPD, such a **Period** element represents a 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.

Table 4: Semantics of Period Element

Element or Attribute Name	Use	Description
Period		specifies the information for a single Period.
@xlink:href	O	specifies a reference to an external **Period** element
@xlink:actuate	O default: onRequest	specifies the processing instructions, which can be either "onLoad" or "onRequest". This attribute must not be present if the @xlink:href attribute is not present
@id	O	specifies a unique identifier for this Period within the Media Presentation. The @id of the Period shall remain unchanged over an MPD update.
@start	O	specifies the start time of the Period.
@duration	O	specifies the duration of the Period.
@segmentAlignmentFlag	OD Default: false	When set to "true", specifies that all presentation start and end times of media components of any particular media type are temporally aligned in all Segments, except possibly the last one, across all Representations with the same value of the @duration attribute on Representation level in this Period.
@bitstreamSwitchingFlag	OD Default: false	When this flag is set to "true", the following apply for any Group in the Period: - All Representations in the Group have the same number of Media Segments - Let *M* be the number of Media Segments in each Representation in the Group. Let $R_1, R_2, ..., R_N$ be all the Representations within the Group. Let $S_{i,j}$ be the j^{th} media segment in the i^{th} Representation (i.e. R_i). The concatenation of any Initialisation Segment, if present, in the Group, with $S_{i(1),1}$, $S_{i(2),2}$, ..., $S_{i(m),m}$, 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, inclusive, and m is less than or equal to M, starting from the Initialization Segment followed by the Media Segments in the order listed, results in a syntactically valid bitstream according to the specific media format that is also semantically correct (i.e. if the concatenation is played, the media content within this Period is correctly presented). This flag shall not be set to "true" when @segmentAlignmentFlag is set to "false". More detailed rules may be defined for specific Initialisation and Media Segment formats.
SegmentInfoDefault	0...1	specifies default Segment information For more details see clause 8.4.4.
AdaptationSet	0...N	specifies an Adaptation Set. For more details see subclause 8.4.3.3.
Representation	0…N	specifies a Representation. For more details see subclause 8.4.3.4.

Element or Attribute Name	Use	Description
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 @.		

Table 5: Syntax of `Period` Element

```
<!-- Period of a presentation -->
<xs:complexType name="PeriodType">
    <xs:sequence>
        <xs:element name="SegmentInfoDefault" type="SegmentInfoDefaultType" minOccurs="0"/>
        <xs:element name="Representation" type="RepresentationType" minOccurs="0"
            maxOccurs="unbounded"/>
        <xs:element name="AdaptationSet" type="AdaptationSetType" 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="segmentAlignmentFlag" type="xs:boolean" default="false"/>
    <xs:attribute name="bitStreamSwitchingFlag" type="xs:boolean" default="false"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

8.4.3 Representation, Groups and Adaptation Sets

8.4.3.1 Overview

Each Period consists of one or more Groups. Groups consist of Adaptation Sets which each consists of one or more Representations.

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.

Representations are arranged into Adaptation Sets according to their language, media type(s), viewpoint, role, and rating. Representations shall appear in the same Adaptation Set if and only if they share common values for all of these properties. All Representations contained in one Adaptation Set represent media components that are considered to be automatically switchable. The media types(s) property of a Representation in this definition is the set of different media types (i.e. audio, video, text) contained within that Representation.

Each **`AdaptationSet`** element describes an Adaptation Set as defined in subsection 8.4.3.3. **`AdaptationSet`** elements are contained in a **`Period`** element. Each **`Representation`** element describes a Representation as defined in 8.4.3.4. **`Representation`** elements are either contained in the **`AdaptationSet`** element or directly in a **`Period`** element.

`AdaptationSet` and **`Representation`** elements share some common elements and attributes as collected in 8.4.3.2. Note that these elements and attributes may also be common to **`SubRepresentation`** elements. **`SubRepresentation`** elements describe Sub-Representations as defined in clause 8.4.3.5.

For each **`AdaptationSet`** element,

- all contained Representations comprise one Adaptation Set. No other Representation not contained in this **`AdaptationSet`** element is part of this Adaptation Set.

- the attribute @group and @lang as well as the elements **Viewpoint**, **Role**, and **Rating** shall only be present in the **AdaptationSet** element, but not in the **Representation** or **SubRepresentation** element.

- any of the attributes @width, @height, @startWithRAP, @codecs, @maxPlayoutRate, @codingDependency, and @frameRate that is specified in both the **Representation** element and the containing **AdaptationSet** element shall have the same value in both.

- all contained Representations are part of one Group. If the **AdaptationSet**@group attribute is present, then all Representations contained in this **AdaptationSet** element are in the Group indicated by the value of this **AdaptationSet**@group attribute. If the **AdaptationSet**@group attribute is not present all contained Representations are part of Group zero.

For each **Representation** element that is directly contained in a **Period** element,

- the described Representation is assigned to the Group given by the value of the **Representation**@group attribute, if present. If the **Representation**@group attribute is not present, then the Representation is assigned to Group zero;

- the described Representations shall be arranged into Groups according to their media type(s), viewpoint, role, and rating regardless whether associated attributes or elements are present, i.e. **Representation**@group for one Representation shall be equal to **Representation**@group for another Representation, if and only if each of these properties are the same for the two Representations.

- all Representations with the same **Representation**@group attribute and the same **Representation**@lang attribute, if present, form an Adaptation Set.

For future releases or proprietary extensions, if different semantic or new non-compatible features are defined, such that HTTP Streaming clients can not properly render a Representation ignoring those extensions, then a new **Representation** element shall be defined, e.g. a new element **Representation2010**. In this example, an MPD may mix both **Representation** and **Representation2010** elements.

8.4.3.2 Common Attributes and Elements

The elements **AdaptationSet, Representation** and **SubRepresentation** have assigned common attributes and elements.

The attributes @width, @height, @lang, @mimeType, @group, @startWithRAP, @codec, @maxPlayoutRate, @codingDependency, and @frameRate may be present in all three elements. The semantics of these attributes are provided Table 6. The XML-syntax is provided in Table 7.

The 'Use' column in Table 6 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.

Table 6: Common Adaptation Set, Representation and Sub-Representation and Attributes and Elements

Element or Attribute Name	Use	Description
Common attributes for **Adaptationset**, **Representation**, and **SubRepresentation**		
@width	O	Specifies the horizontal visual presentation size of the video media type in pixel.
@height	O	Specifies the vertical visual presentation size of the video media type in pixel.
@frameRate	O	Specifies the output frame rate or the output field rate of the video media type in frames per second. If the frame rate is varying, the value is the average frame rate over the entire duration of the described Representation.
@lang	O	Specifies the language code(s) according to IETF RFC 5646 [13].

Element or Attribute Name	Use	Description
		NOTE: Multiple language codes may be declared as a white-space separated list and indicate a preference suitable for any of the indicated languages. For a full indication of what media is offered under each language, the Initialisation Segment or a Media Segment may have to be accessed.
@mimeType	M	Specifies the MIME type of the Initialisation Segment, if present; if the Initialisation Segment is not present it provides the MIME type of the first Media Segment. For the 3GPP file format, the MIME type shall be provided according to RFC 4281 [14].
@codecs	M	Specifies the codecs parameter specifying the media types. The codec parameters shall also include the profile and level information where applicable. The contents of this attribute shall conform to either the simp-list or fancy-list productions of RFC 4281 clause 3.2, without the enclosing DQUOTE characters. The codec identifier for the media format, mapped into the name space for codecs as specified in RFC 4281, clause 3.3 shall be used.
@group	O	Specifies the group.
@startWithRAP	O	When 'true', specifies that all Segments in the described Representations start with a RAP (both in terms of data and in terms of presentation time). The presentation time of the RAP shall either be provided explicitly by the Segment Index or, if the @segmentAlignmentFlag is true, may be inferred from the presentation time of the last sample of the previous segment.
@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.
@codingDependency	O	When present and "true", it specifies that for all media types, there is at least one access unit that depends on one or more other access units for decoding. When present and "false", for any media type, there is no access unit that depends on any other access unit for decoding (e.g. for video all the pictures are intra coded). When not present, there may or may not be coding dependency between access units.
ContentProtection	0 … N	Specifies information about the use of content protection for the described Representations. For more details, refer to 8.6.3.1 and 8.6.3.2.
Role	0 … N	Specifies information on Role annotation scheme For more details refer to 8.6.3.1 and 8.6.3.3.
Rating	0 … N	Specifies information about Content rating scheme For more details refer to 8.6.3.1 and 8.6.3.4.
Viewpoint	0 … N	Specifies information Content View Point annotation scheme For more details refer to 8.6.3.1 and 8.6.3.5.
Legend: For attributes: M=Mandatory, O=Optional. For elements: <minOccurs>..<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

Table 7: XML-Syntax of Common Group and Representation and Attributes and Elements

```
<!-- RepresentationBase type; extended by other Representation-related types -->
<xs:complexType name="RepresentationBaseType" abstract="true">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="ContentProtection" 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:choice>
    <xs:attribute name="width" type="xs:unsignedInt"/>
    <xs:attribute name="height" type="xs:unsignedInt"/>
    <xs:attribute name="frameRate" type="xs:double"/>
    <xs:attribute name="lang" type="LangVectorType"/>
    <xs:attribute name="mimeType" type="xs:string"/>
    <xs:attribute name="codecs" type="xs:string"/>
    <xs:attribute name="group" type="xs:unsignedInt"/>
    <xs:attribute name="startWithRAP" type="xs:boolean"/>
    <xs:attribute name="maxPlayoutRate" type="xs:double"/>
    <xs:attribute name="codingDependency" type="xs:boolean"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- Type for space delimited list of language codes -->
<xs:simpleType name="LangVectorType">
    <xs:list itemType="xs:language"/>
</xs:simpleType>
```

8.4.3.3 Adaptation Set

Adaptation Sets are typically be described by the **AdaptationSet** element. This element supports the description of ranges for the @bandwidth, @width, @height and @frameRate attributes defined for the **Representation** element, which provide a summary of all values for all the Representations within this Adaptation Set. The Representations associated with an **AdaptationSet** element shall not contain values outside the ranges documented for that Adaptation Set.

The semantics of the attributes and elements within a **AdaptationSet** element are provided in Table 8. The XML-syntax of the Period type is provided in Table 9.

Table 8: Semantics of AdaptationSet element

Element or Attribute Name		Use	Description
AdaptationSet			Adaptation Set description
	@xlink:href	O	specifies reference to external **AdaptationSet** element
	@xlink:actuate	OD default: 'onRequest'	specifies the processing instructions, which can be either "onLoad" or "onRequest".
	@id	O	specifies unique identifier for this Adaptation Set within the Period.
	CommonAttributesElements	-	Common Adaptation Set, Representation and Sub-Representation Attributes and Elements (see clause 8.4.3.2)
	@minBandwidth	O	specifies minimum bandwidth value in all Representations in this Adaptation Set.
	@maxBandwidth	O	specifies maximum bandwidth value in all Representations in this Adaptation Set.
	@minWidth	O	specifies minimum width value in all Representations in this Adaptation Set.
	@maxWidth	O	specifies maximum width value in all Representations in this Adaptation Set.
	@minHeight	O	specifies minimum height value in all Representations in this Adaptation Set.
	@maxHeight	O	specifies maximum height value in all Representations in this Adaptation Set.

Element or Attribute Name	Use	Description
@minFrameRate	O	specifies minimum frame rate value in all Representations in this Adaptation Set.
@maxFrameRate	O	specifies maximum frame rate value in all Representations in this Adaptation Set.
@segmentAlignmentFlag	O	If present, overrides **Period**@segmentAlignmentFlag in and specifies the value for all Representations in this Adaptation Set.
@bitStreamSwitchingFlag	O	If present, overrides **Period**@bitStreamSwitchingFlag and specifies the value for all Representations in this Adaptation Set.
SegmentInfoDefault	0...1	If present, replaces any possibly present **Period**@**SegmentInfoDefault**. For more details refer to clause 8.4.4.
Representation	1 … N	See subclause 8.4.3.4.

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***

Table 9: XML-Syntax of `Group` element

```
    <!-- Group to contain information common to an adaptation set;
        extends RepresentationBaseType -->
    <xs:complexType name="AdaptationSetType">
        <xs:complexContent>
            <xs:extension base="RepresentationBaseType">
                <xs:sequence>
                    <xs:element name="SegmentInfoDefault" type="SegmentInfoDefaultType"
                        minOccurs="0"/>
                    <xs:element name="Representation" type="RepresentationType" 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="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="xs:double"/>
                <xs:attribute name="maxFrameRate" type="xs:double"/>
                <xs:attribute name="segmentAlignmentFlag" type="xs:boolean"/>
                <xs:attribute name="bitStreamSwitchingFlag" type="xs:boolean"/>
                <xs:anyAttribute namespace="##other" processContents="lax"/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

8.4.3.4 Representation

Representations are described by the **Representation** element. A Representation is one of the alternative choices of the media content or a subset thereof typically differing by the encoding choice, e.g. by bitrate, resolution, language, codec, etc.

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.

A Representation consists of one or more Segments.

Each Representation either contains an Initialisation Segment or each Media Segment in the Representation is self-initialising.

Each Representation includes one or more media components, where each media component is an encoded version of one individual media type of continuous media such as audio, video, or timed text. Media components are time-continuous across boundaries of consecutive Media Segments within one Representation. The timing within each Representation is relative to the *PeriodStart* time or the Period that contains this Representation.

The semantics of the attributes and elements within a Representation are provided in Table 10. The XML-syntax of the **`Representation`** element is provided in Table 11.

Table 10: Semantics of `Representation` element

Element or Attribute Name	Use	Description
`Representation`		This element contains a description of a Representation.
`@id`	M	specifies a unique identifier for this Representation within the Period. The string shall only contain characters that permit to form a valid HTTP-URL according to RFC 1738.
`@bandwidth`	M	specifies the minimum bandwidth of a hypothetical constant bitrate channel in bits per second (bps) over which the Representation (i.e. the collection of all Segments of a Representation) can be continuously delivered such that a client, after buffering for exactly `@minBufferTime` when accessing a Representation at any RAP can be assured of having enough data for continuous playout.
`@qualityRanking`	O	specifies a quality ranking of the Representation relative to other Representations in the Adaptation Set. Lower values represent higher quality content. If not present then the ranking is undefined.
`CommonAttributesElements`	-	specifies common Adaptation Set, Representation and Sub-Representation attributes and elements (see clause 8.4.3.2)
`SubRepresentation`	0 … N	specifies rovides information about a sub-representation that is embedded in the containing Representation. For more details see clause 8.3.4.5.
`SegmentInfo`	1	specifies Segment information. For more details see clause 8.4.4.
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 @, List of elements and attributes is in ***italics bold***		

Table 11: XML-Syntax of `Representation` element

```
    <!-- A Representation of the presentation content for a specific Period -->
    <xs:complexType name="RepresentationType">
        <xs:complexContent>
            <xs:extension base="RepresentationBaseType">
                <xs:sequence>
                    <xs:element name="SubRepresentation" type="SubRepresentationType"
minOccurs="0"/>
                    <xs:element name="SegmentInfo" type="SegmentInfoType" minOccurs="0"/>
                    <xs:any namespace="##other" processContents="lax" minOccurs="0"
                        maxOccurs="unbounded"/>
                </xs:sequence>
                <xs:attribute name="id" type="xs:string" use="required"/>
                <xs:attribute name="bandwidth" type="xs:unsignedInt" use="required"/>
                <xs:attribute name="qualityRanking" type="xs:unsignedInt"/>
                <xs:anyAttribute namespace="##other" processContents="lax"/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
```

8.4.3.5 Sub-Representation

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

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.

Many attributes and elements that are used for the description of Representations are also applicable for the description of Sub-Representations.

In case the `@level` attribute is present in the **`SubRepresentation`** element, then the Initialisation Segment and/or the Media Segments shall provide sufficient information such that the data can be easily accessed through HTTP partial GET requests.

NOTE: If the `@level` attribute is absent, then the **`SubRepresentation`** element is only used as a more detailed descriptor for components that are embedded in the Representation.

The semantics of the attributes and elements within a Sub-Representation are provided in Table 12. The XML-syntax of the Sub-Representation element is provided in Table 13.

Table 12: Semantics of `SubRepresentation` element

Element or Attribute Name	Use	Description
`SubRepresentation`		This element specifies a Sub-Representation. If `@level` attribute present, a Subsegment Index is available for each Media Segment in the containing Representation.
`@level`	O	Specifies the sub-representation level.
`@dependencyLevel`	O	specifies a whitespace-separated list of `@level` attributes indicating all Sub-Representations within this Representation that this Sub-Representation depends on in the decoding process.
`@bandwidth`	O	Identical to the `@bandwidth` definition in Representation, but applied to this Sub-Representation. This attribute shall be present in case the `@level` attribute is present.
`CommonAttributesElements`	-	Common Adaptation Set, Representation and Sub-Representation attributes and elements (see clause 8.3.4.2)
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***		

Table 13: XML-Syntax of `SubRepresentation` element

```
<!-- SubRepresentation of the presentation content for a specific Period;
    extends RepresentationBaseType -->
<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:anyAttribute namespace="##other" processContents="lax"/>
        </xs:extension>
    </xs:complexContent>
</xs:complexType>

<!-- Type for space delimited list of strings -->
<xs:simpleType name="UIntVectorType">
    <xs:list itemType="xs:unsignedInt"/>
</xs:simpleType>
```

8.4.4 Segments and Segment Information

8.4.4.1 General

A Segment is defined as a unit that can be referenced by an HTTP-URL included in the MPD, where an HTTP-URL is defined as an <absolute-URI> according to RFC 3986 [17] , Subclause 4.3, with a fixed scheme of 'http://' or https://, possibly restricted by a byte range if the **`Url`**`@range` attribute is provided. The byte range is expressed as a `byte-range-set` as defined in RFC 2616 [9], clause 14.35.1. It is restricted to a single expression identifying a contiguous range of bytes.

The Initialisation Segment contains initialisation information for accessing the Representation. The Initialisation Segment shall not contain any media data. If the MPD contains information that is also documented in the Initialisation Segment, then the information in the MPD shall be such that no contradication between these two values occur.

A Media Segment contains media components that are either described within this Media Segment or described by the Initialisation Segment of this Representation. In addition, a Media Segment:

- is assigned an HTTP-URL, possibly restricted by a byte range.
- is explicitly or implicitly assigned a start time relative to the start of the Representation provided by the MPD. The client can therefore download the appropriate Segments in regular play-out mode or after seeking. The start time shall be drift-free between the time indicated in the MPD and internal clock of the Media Segments, i.e. the accuracy of the start time documented in the MPD relative to the internal clock does not depend on the position of the Segment in the Representation.
- should contain at least one representation access point (RAP). Note that certain profiles may mandate the presence of at least one RAP for each media segment.
- should provide information how to access the Media Presentation within this Segment, e.g. exact timing, byte index. There is no requirement that a Media Segment starts with a RAP, but it is possible to signal in the MPD that all Segments within a Representation start with a RAP. The first Media Segment of a Representation shall always start with a RAP.
- 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 RAP. The time-accuracy is necessary for a client to seamlessly switch Representations and jointly present multiple Representations.
- may contain indexing information for randomly accessing subsegments of the Segment by using partial HTTP GET requests. Indexing information may also be provided in separate Index Segments. A generic syntax and semantic for indexing is for example provided by the Segment Index ("`sidx`") box in TS 26.244 [4].
- may contain additional subsegment indexing information for accessing different levels of subsegments in a media segment. A generic syntax and semantic for sub-segment indexing is for example provided by the Subsegment Index ("`ssix`") box in Annex G.2.
- if first in a Representation, shall be assigned presentation start time 0 in the Media Presentation timeline relative to the Period start time. If the first sample of any of the media components in a Representation has a Media Presentation time relative to the Period start time greater than 0, then this presentation time shall be signalled in the Media Segment. No sample of any of the media components shall have Media Presentation time relative to the Period start time smaller than 0.

Each **`Representation`** element shall contain at most one **`SegmentInfo`** element that together with the possibly present elements **`Period.SegmentInfoDefault, AdaptationSet.SegmentInfoDefault`** and **`MPD.BaseURL`** as well as a document "base URI" shall contain sufficient information to determine the *Segment Information*.

If at least one of the elements **`Period.SegmentInfoDefault`** or **`AdaptationSet.SegmentInfoDefault`** are present, then combination rules with the information in **`SegmentInfo`** apply to obtain a derived structure for the **`SegmentInfo`** element. These rules are provided in 8.4.4.2.

The rules to determine the *Segment Information* based on a **SegmentInfo** element are provided in 8.4.4.3. These rules apply to the derived **SegmentInfo** element after application of the rules in 8.4.4.2.

The semantics of the attributes and elements for the Segment Default element and Segment Information element are provided in Table 14 and Table 15, respectively. The XML-syntax of the Segment Information is provided in Table 16.

Table 14: Semantics of SegmentInfoDefault element

Element or Attribute Name	Use	Description
SegmentInfoDefault		specifies default Segment information
@duration	O	specifies the default duration of Media Segments
@startIndex	O	specifies the default start index
@sourceURLTemplate	O	specifies the media segment URL template on the level where the segment information is placed.
InitialisationSegmentURL	0…1	specifies default for Initialisation Segment URL
BaseURL	0…N	specifies the Base URL element to be used for reference resolution
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 15: Semantics of SegmentInfo element

Element or Attribute Name	Use	Description
SegmentInfo		specifies segment 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.
@startIndex	O	specifies the index of the first accessible Media Segment in this Representation. In case of on-demand services or in case the first Media Segment of the Representation is accessible, then this value shall not be present or shall be set to 1.
BaseURL	0...N	specifies the Base URL element that can be used for reference resolution
InitialisationSegmentURL	0...1	specifies the element for the Initialisation Segment.
@sourceURL	O	specifies the source string providing the URL. 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 URL. If not present, the resources referenced in the sourceURL are unrestricted. The format of the string shall comply with the format as specified in 0.
UrlTemplate	0 … 1	specifies the element includes attributes to generate a Segment list for the Representation associated with this element.
@sourceURL	O	specifies the string providing the template to create the Media Segment List.
Url	0 ... N	specifies a set of explicit URL(s) for Segments. NOTE: The URL element may contain a byte range.
@sourceURL	O	specifies the source string providing the URL. If not present, then any **BaseURL** element is mapped to the @sourceURL attribute and the @range attribute shall be present.

Element or Attribute Name		Use	Description
	@range	O	specifies the byte range restricting the above URL. If not present, the resources referenced in the sourceURL are unrestricted. The format of the string shall comply with the format as specified in 8.4.4.
SegmentList		0 … N	specifies a list of explicit URL(s) for Segments.
	@xlink:href	O	specifies a reference to external **SegmentList** element
	@xlink:actuate	OD default: "onRequest"	specifies the processing
	@startIndex	O	specifies start index for this segment list.
	Url	0 … N	specifies a Media Segment URL

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 @.

Table 16: XML-Syntax of `SegmentInfoDefault` of `SegmentInfo` element

```
    <!-- Default Segment access information -->
    <xs:complexType name="SegmentInfoDefaultType">
        <xs:sequence>
            <xs:element name="BaseURL" type="BaseUrlType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="InitialisationSegmentURL" type="UrlType" minOccurs="0"/>
            <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attributeGroup ref="SegmentInfoAttrGroup"/>
        <xs:attribute name="sourceURLTemplate" type="xs:string"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- grouping attributes common to SegmentInfo and SegmentInfoDefault -->
    <xs:attributeGroup name="SegmentInfoAttrGroup">
        <xs:attribute name="duration" type="xs:duration"/>
        <xs:attribute name="startIndex" type="xs:unsignedInt" default="1"/>
    </xs:attributeGroup>


    <!-- Segment access information -->
    <xs:complexType name="SegmentInfoType">
        <xs:sequence>
            <xs:element name="BaseURL" type="BaseUrlType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="InitialisationSegmentURL" type="UrlType" minOccurs="0"/>
            <xs:choice minOccurs="0">
                <xs:element name="UrlTemplate" type="UrlTemplateType" minOccurs="0"/>
                <xs:sequence>
                    <xs:element name="Url" type="UrlType" maxOccurs="unbounded"/>
                    <xs:any namespace="##other" processContents="lax" minOccurs="0"
                        maxOccurs="unbounded"/>
                </xs:sequence>
                <xs:element name="SegmentList" type="SegmentListType" minOccurs="0"/>
                <xs:any namespace="##other" processContents="lax" minOccurs="0"
                    maxOccurs="unbounded"/>
            </xs:choice>
        </xs:sequence>
        <xs:attributeGroup ref="SegmentInfoAttrGroup"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- A Segment URL  -->
    <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>

    <!-- A URL template  -->
    <xs:complexType name="UrlTemplateType">
        <xs:sequence>
            <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="sourceURL" type="xs:anyURI"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- SegmentList allows xlink in addition to list of URLs -->
    <xs:complexType name="SegmentListType">
        <xs:sequence>
            <xs:element name="Url" type="UrlType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute ref="xlink:href"/>
        <xs:attribute ref="xlink:actuate" default="onRequest"/>
        <xs:attribute name="startIndex" type="xs:unsignedInt"/>
    </xs:complexType>
```

8.4.4.2 Combination Rules to obtain Derived SegmentInfo Element

Reference resolutions as defined in 8.2.3 and base URL selection as defined in 8.2.4 shall be applied to obtain a base URL value.

The following rules apply for the combination of **`Period.SegmentInfoDefault`**, **`AdaptationSet.SegmentInfoDefault`** and **`Representation.SegmentInfo`** to obtain a derived **`SegmentInfo`** element.

First, an attempt is made to produce a derived **`SegmentInfoDefault`** element:

- If a **`AdaptationSet.SegmentInfoDefault`** is present then this element is the derived **`SegmentInfoDefault`** element.
- If a **`AdaptationSet.SegmentInfoDefault`** is not present, but a **`Period.SegmentInfoDefault`** is present then this **`Period.SegmentInfoDefault`** is the derived **`SegmentInfoDefault`** element.
- If no derived **`SegmentInfoDefault`** element has been produced, there must be a **`Representation.SegmentInfo`** element present and it is used as the derived **`SegmentInfo`** element.

At this point if the derived **`SegmentInfo`** element has been produced, processing is complete. Otherwise, the following applies:

- If no derived **`SegmentInfoDefault`** element has been produced, the MPD is invalid and processing stops.
- If no **`Representation.SegmentInfo`** element is present, then the following applies for mapping the derived **`SegmentInfoDefault`** element to the derived **`SegmentInfo`** element:
  - The value of each element or attribute that is present in the derived **`SegmentInfoDefault`** is assigned to the corresponding element or attribute (with the same name) in the derived **`SegmentInfo`** element. This applies for **`InitialisationSegmentURL`**, `@duration`, and `@startIndex`.
  - The value of a possibly present **`SegmentInfoDefault`**`@sourceURLTemplate` attribute is assigned to value of the **`SegmentInfo.URLTemplate`**`@sourceURL` attribute.
- If a **`Representation.SegmentInfo`** element is present then the following applies for combining it with the derived **`SegmentInfoDefault`** to produce the derived **`SegmentInfo`** element.
  - The value for each element or attribute that is present in the derived **`SegmentInfoDefault`** and not present in **`Representation.SegmentInfo`** is assigned to value of the corresponding element or attribute (with the same name) in the derived **`SegmentInfo`** element. This applies for **`InitialisationSegmentURL`**, `@duration`, and `@startIndex`.

- If **SegmentInfo**.**URLTemplate**@sourceURL is not present, but **SegmentInfoDefault**@sourceURLTemplate attribute is present, then the value of this latter attribute is assigned to value of the attribute **SegmentInfo**.**URLTemplate**@sourceURL.

- If **SegmentInfoDefault** and **Representation.SegmentInfo** both contain one or more base URLs (specified using the **BaseURL** element), then this creates a level for reference resolution as defined in 8.2.3.

8.4.4.3 Segment Information based on Derived SegmentInfo element

8.4.4.3.1 Overview

The *Segment Information* is described in a **SegmentInfo** element (or in an derived structure after application of the rules in 8.4.4.2) and provides the following information:

- the presence or absence of Initialisation Segment information
- the HTTP-URL and byte range for each accessible Segment in each Representation,
- all valid segment URLs declared by the containing MPD
- for live services, the *segment availability start time* and *segment availability end time* of each Segment relative to the sum of the value of the **MPD**@availabilityStartTime and the *PeriodStart* time,
- the approximate media start time of a Media Segment in the media presentation timeline within the Period,

Any Segment URL, regardless whether provided explicitly or for example derived through Template-based Segment URL construction as defined in 8.4.4.4 may be relative or absolute. The resulting URLs shall be resolved with respect to a derived **SegmentInfo**.**BaseURL** element as defined in 8.2.3. Furthermore, if a derived **SegmentInfo**.**BaseURL** element is present, then alternative base URL selection as defined in 8.2.4 shall be applied.

The derivation of Initialisation and Media Segment Information is provided in subclause 8.4.4.3.2 and 8.4.4.3.3, respectively.

8.4.4.3.2 Initialisation Segment Information

Each Representation has assigned at most one Initialisation Segment. The Initialisation Segment is typically processed to initialise the media engines for enabling play-out of Media Segments of the containing Representation.

The presence of an Initialisation Segment is indicated either

- by the presence of the **InitialisationSegmentURL** element that contains the URL to the Initialisation Segment, possibly restricted by a byte range, or
- by the presence of **UrlTemplate**@sourceURL even if the **InitialisationSegmentURL** element is absent. In this case the Template-based Segment URL construction in 8.4.4.4 shall be applied with Index set to 0 to obtain the URL to the Initialisation Segment.

If no Initialisation Segment URL is present for a Representation then each Media Segment within the Representation shall be self-initialising.

Any Initialisation Segment shall be a valid segment.

For live services, the segment availability start time of the Initialisation Segment is the the sum of the value of the **MPD**@availabilityStartTime and *PeriodStart* time and the segment availability end time of the Initialisation Segment is the largest segment availability end time of any Media Segment in this Representation. For segment availability for media segments refer to clause 8.4.4.3.3.

8.4.4.3.3 Media Segment Information

Each Representation has assigned a list of consecutive Media Segments. Each entry in the list of a media segment has assigned the following parameters:

- a valid Media Segment URL, possibly restricted by a byte range
- the index of the Media Segment in the Representation
- the approximate presentation start time of the Media Segment in the Representation
- the approximate presentation duration of the Media Segment

The following information in the **SegmentInfo** element determines these parameters: **UrlTemplate**, **Url** and **SegmentList** elements as well as @duration and @startIndex attribute.

The list of Media Segments shall contain at least one entry. The derived **SegmentInfo** element shall contain exactly one of the following choices to determine the Media Segment URL and the Index of the Media Segment:

- one **UrlTemplate** element: In this case the Template-based Segment URL construction in 8.4.4.4 shall be applied with the Index of the Media Segment in the media segment list. The first index in the list is determined by the value of the **SegmentInfo**@startIndex attribute, if present, or is 1 in case this attribute is not present.
- one or more **Url** elements: In this case the **Url** is directly assigned to the Media Segment URL. The first index in the list is determined by the value of the **SegmentInfo**@startIndex attribute, if present, or is 1 in case this attribute is not present. The last index is one less than the sum of the first index and the number of list entries.
- one or more **SegmentList** elements that each contains a list of **Url** elements for a consecutive list of Media Segment URLs. The first index in the list documented by any **SegmentList** element is determined by the value of the **SegmentList**@startIndex attribute, if present, or is identical to **SegmentInfo**@startIndex in case this attribute is not present. If the **SegmentInfo**@startIndex is not present either, the value is set to 1. The sequence of multiple **SegmentList** elements within a Representation shall result in Media Segment List with consecutive indices.
- none of the above: In this case the same procedure shall be followed as for a single **Url** element containing an empty @sourceURL attribute.

For the derivation of the approximate presentation start time and duration of each Media Segment in the list of media segments, the index of the media segment *Index* and the following information are used

- If the @duration attribute is not present, then the Representation shall contain exactly one Media Segment. The presentation start time is 0 and the duration is obtained as it would be the last Media Segment in the Representation (see below for more details).
- If @duration attribute is present, then the approximate presentation start time of the Media Segment is determined as (*Index*-1) times the value of the duration of the attribute @duration. The duration of the Media Segment is determined as the value of the attribute @duration except for the duration of the last Media Segment (see below for more details).
- To determine the duration of 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 add **Period**@duration, or next **Period**@start may be present.

The start time is relative to the start of the Representation provided by the MPD. The start time is approximate and does not reflect the exact media time. However, the start time shall be drift-free between the time indicated in the MPD and internal clock of the Media Segments, i.e. the accuracy of the start time documented in the MPD relative to the internal clock does not depend on the position of the Segment in the Representation.

For live services, the segment availability start time of a Media Segment is the sum of the value of the **MPD**@availabilityStartTime , the *PeriodStart* time and the start time of the Media Segment in the Representation. The segment availability end time of a Media Segment is the sum of the segment availability start time, the duration of the Media Segment and the value of the attribute **MPD**@timeShiftBufferDepth.

The MPD shall include URL information for all segments with an availability start time less than both the end of the presentation and the sum of the latest time at which this version of the MPD is available on the server and the **MPD**@minimumUpdatePeriodMPD.

8.4.4.4 Template-based Segment URL Construction

The derived **SegmentInfo** element may contain **UrlTemplate** element. The **UrlTemplate**@sourceURL attribute represents a string that contains one or more of the identifiers as listed in Table 17. The attributes shall contain the *$Index$* identifier.

A sub-string "$<Identifier>$" names a substitution placeholder matching a mapping key of "<Identifier>". In the request URL, the substitution placeholder shall be replaced by the substitution parameter as defined in Table 17. Substitution is performed left to right and identifier matching is case-sensitive. Unrecognized identifiers cause the URL formation to fail. In this case it is expected that the 3GP-DASH client ignores the entire containing **Representation** element and the processing of the MPD continues as if this **Representation** element was not present.

It is the responsibility of the content author that the application of the substitution process results in valid Segment URLs.

Strings outside identifiers shall only contain characters that permit to form a valid HTTP-URL according to RFC 1738 [19].

Table 17: Identifiers for URL Templates

$<Identifier>$	Substitution parameter
$$	Is an escape sequence, i.e. "$$" is replaced with a single "$"
$RepresentationID$	This identifier is substituted by the attribute **Representation**@id of the containing Representation.
$Index$	This identifier is substituted by the *Index* of the corresponding Segment. For an example 3GP-DASH client using this identifier to construct the list Media Segment URLs, refer to A.3.

8.5 MPD Update

8.5.1 General

If the **MPD**@type is set to 'Live', the MPD may be updated during the Media Presentation.

In this case the MPD shall be made accessible at all locations specified in any present **MPD.Location** element or, if none is present, at the same location as the initial MPD. If the client fetches the MPD using HTTP, the client should use conditional GET methods as specified in RFC 2616 [9], clause 9.3 to reduce unnecessary network usage in the downlink.

When the MPD is updated 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**@minimumUpdatePeriodMPD is present, updates to the MPD are expected, but no earlier than indicated by the value of **MPD**@minimumUpdatePeriodMPD as follows.

- If the attribute **MPD**@minimumUpdatePeriodMPD is not present, no update to the MPD is expected and the attribute **MPD**@mediaPresentationDuration shall be present.

- If the attribute **MPD**@minimumUpdatePeriodMPD is present, let *Texp(i)* be the sum of the value of **MPD**@minimumUpdatePeriodMPD and the wall-clock time at which the *i*-th version of the MPD is updated (and replaced with the (*i*+1)-th version). If the attribute **MPD**@minimumUpdatePeriodMPD is not present let *Texp(i)* be the end of the Media Presentation, i.e. the sum of the value of the **MPD**@availabilityStartTime and the value of **MPD**@mediaPresentationDuration.

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*). The actual duration of this segment is not known by the client until this segment or the next update of MPD is fetched and this duration may be less than the normal segment duration if it is the last Segment of the Representation in this Period.

NOTE:

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

2) The second condition above ensures that sufficient media is contained in each Representation to present up to media presentation time *Texp*(*i*).

8.5.2 Media Presentation Description Delta

If the **DeltaSupport** element is present in the **MPD** element, the content provider indicates that MPD delta files, as defined in this clause, are supported on the server. The URI of the MPD delta is provided in **DeltaSupport** @sourceURL. The **DeltaSupport** @availabilityDuration element, if present, indicates that the MPD delta file referenced by the URI is available for at least the value of the @availabilityDuration attribute (after this time, the server may redirect the client to the full MPD). If **DeltaSupport** @availabilityDuration is not present, then no information is conveyed about the availability of the MPD delta. If a client request for an MPD delta file results in an error, the client should request a full MPD.

The semantics of the attributes within the **DeltaSupport** element are provided in Table 18. The XML-syntax of **DeltaSupport** element is provided in Table 19.

Table 18: Semantics of DeltaSupport element

Element or Attribute Name	Use	Description
DeltaSupport		If present, this element indicates that MPD delta files are supported by the server.
@sourceURL	M	The source string providing the URL of the MPD delta. The URL may be relative to any **BaseURL** on MPD level and reference resolution according to clause 8.2.3 shall be applied.
@availabilityDuration	O	When provided, indicates the duration that the server guarantees the availability of the MPD delta file referenced in @sourceURL after the MPD has been updated. After that the client may be redirected to the full MPD.
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 19: XML-Syntax of `DeltaSupport` element

```
<!--DeltaSupport for the MPD -->
<xs:complexType name="DeltaSupportType">
    <xs:sequence>
        <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="sourceURL" type="xs:anyURI" use="required"/>
    <xs:attribute name="availabilityDuration" type="xs:duration"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

An MPD delta is a text file that shall include the delta between the MPD that references it and the latest provided MPD. Note that the value of `@sourceURL` in successive MPDs is necessarily different because it is impossible for the delta between two different MPDs and the most recent MPD to be the same.

The output format consists of one or more hunks of differences. The changes are in decreasing line number order. The hunks format look like this:

```
change-command
to-file-line
to-file-line...
.
```

There are three types of change commands `change-command`. Each consists of a line number or comma-separated range of lines in the first file and a single character indicating the kind of change to make. All line numbers are the original line numbers in the file. The types of change commands and the instructions are provided in Table 20.

Table 20: Change commands and the instructions for delta MPD files

Change command	Instruction	Example
*l*a	Add text from the second file after line *l* in the first file.	"8a" means to add the following lines after line 8 of file 1
*r*c	Replace the lines in range *r* in the first file with the following lines. Like a combined add and delete, but more compact.	"5,7c" means change lines 5–7 of file 1 to read as the text file 2.
*r*d	Delete the lines in range *r* from the first file.	"5,7d" means delete lines 5–7 of file 1.
NOTE: This is the format supported by the GNU diff utilities, see http://www.gnu.org/software/diffutils/manual/#Detailed-ed		

Regardless of the presence of a **`DeltaSupport`** element, the full MPD shall always be available to clients for regular MPD updates as defined in clause 8.5.1. MPD Delta related procedures are optional at the client.

8.6 Additional Media Presentation Information

8.6.1 Introduction

The MPD, Periods, Adaptation Sets, Representations and Sub-Representations may have assigned descriptors for describing the content or other elements in the MPD. This clause specifies this descriptive information.

8.6.2 Program Information

Descriptive information on the program may be provided for each period within the **`ProgramInformation`** element. Attributes and elements are provided to specify a URL for additional information, the title, the source of the program, and some copyright information.

The semantics of the attributes within the **`ProgramInformation`** element are provided in Table 21. The XML-syntax of **`ProgramInformation`** element is provided in Table 22.

Table 21: Semantics of `ProgramInformation` element

Element or Attribute Name	Use	Description
`ProgramInformation`		specifies descriptive information about the program
`@moreInformationURL`	O	If specified, this attribute contains an absolute URL which provides more information about the Media Presentation in this Period.
`Title`	0...1	specifies a 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.

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 22: XML-Syntax of `ProgramInformation` element

```
<!-- Program information for a presentation -->
<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="moreInformationURL" type="xs:anyURI"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

8.6.3 Descriptors

8.6.3.1 General

The MPD may contain descriptors that are all in the same format as defined in this clause. The description elements are all structured in the same way, namely they contain a `@schemeIdUri` attribute to identify the scheme and an optional attribute `@value` attribute. The `@schemeIdUri` provides a URI to identify the scheme. The definition of this element is specific to the scheme employed. The scheme may be a URN or a URL.

One set of descriptors are content descriptors. In this case, each Representation or per default each Adaptation Set may have assigned certain content description elements. Content Descriptions on Adaptation Sets should be applied such that the information is sufficient to judge whether at least one Representation contained in this Adaptation Set can be accessed. In this version of the specification, specific elements for content description are defined for:

- **`ContentProtection`***:* content protection scheme element, see clause 8.6.3.2.
- **`Role`***:* role annotation scheme element, see clause 8.6.3.3.
- **`Rating`***:* content rating scheme element, see clause 8.6.3.4.
- **`Viewpoint`***:* content viewpoint scheme element, see clause 8.6.3.5.

Other descriptors defined in this specification are

- **`QualityReporting`***:* quality reporting scheme, see clause 10.6.

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 descriptors with appropriate scheme information. This specification defines descriptor schemes in Annex C.

The **`Descriptor`** element provides a flexible mechanism for DASH content authors to annotate and extend the Adaptation Set and Representation elements for the purposes of content protection, role, ratings and viewpoint support.

DASH applications that may define 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 the any element/attribute extensibility in a separate namespace can be used and then may contain the required elements and attributes to provide information specific to the scheme.

DASH applications defining a Scheme Identifier may define (i) a text string to be included in the `@value` attribute, or (ii) a structured extension of the element or (iii) both.

Two elements of type **`DescriptorType`** are *equivalent*, if the element name, the `@schemeIdUri` and the `@value` are equivalent. If the `@schemeIdUri` is a URN, then equivalence refers to lexical equivalence as defined in clause 5 of RFC 2141. If the `@schemeIdUri` is a URI, then equivalence refers to equality on a character-for-character basis as defined in clause 6.2.1 of RFC 3986 [17]. For the `@value` XML-string matching shall be used for determining equivalence. If the `@value` attribute is not present, equivalence is determined by the equivalence for `@schemeIdUri` only.

The semantics of the attributes within a Generic Descriptor element are provided in Table 23. The XML-syntax of a Generic Descriptor element is provided in Table 24. The specific descriptors follow these syntax and semantics.

Table 23: Semantics of generic `Descriptor` element

Element or Attribute Name	Use	Description
`Descriptor`		This element provides information about the use of description.
`@schemeIdUri`	M	Provides a URI to identify the scheme. The definition of this element is specific to the scheme employed for content description. The URI may be a URN or a URL. 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	This attribute provides 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.
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 24: XML-Syntax of generic `Descriptor` element

```
<!-- Generic named descriptive information -->
<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" use="optional"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

8.6.3.2 Content Protection

For the element **ContentProtection** the @schemeIdUri attribute is used to identify the content protection schemes employed. 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). Scheme-specific information can also be provided in the Initialization Segment(s) using the appropriate file format primitives instead of, or in addition to, the **ContentProtection** element. The client may have to receive and analyze the protected content (typically only the Initialization Segment, if present), before it can determine whether it has already acquired a license and/or key for accessing the protected content, or to determine from where it can acquire a missing license and/or key, in case this information is not available from the **ContentProtection** element.

When the **ContentProtection** element is not present the content shall neither be encrypted nor 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.

8.6.3.3 Role

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

Groups and Representations containing different **Role** elements indicate that the contained Representations contain different media components and therefore 3GP-DASH clients should not automatically switch Representations with such differing **Role** elements. One Group or Representation may have assigned multiple roles even within the same scheme. Also in case of multiplexed Representations, the role may describe only one of the components.

Specifically this specification defines a scheme in Annex C.2 of this specification.

8.6.3.4 Rating

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

A Rating specifies that content is suitable for presentation to audiences for which that rating is known to be appropriate, or for unrestricted audiences.

NOTE: An audience with a rating restriction is intended to not be presented content that has associated ratings, unless at least one scheme is recognized as indicating that the content is appropriate to that audience.

8.6.3.5 Viewpoint

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

Groups and Representations containing non-equivalent **Viewpoint** elements indicate that the contained Representations contain different media components (including other media types than video) and therefore 3GP-DASH clients should not automatically select or switch Representations with such non-equivalent **Viewpoint** elements.

Representations with equivalent **Viewpoint** elements and appearing in different Groups are intended to be presented together. This handling should be applied equally for recognised and unrecognised @schemeIdUri values.

9 DASH - Usage of 3GPP File Format

9.1 Introduction

3GPP Dynamic Adaptive Streaming over HTTP uses many elements of fragmented 3GP files to define the segment formats. This provides segments according to the requirements defined in clause 8.4.4.1 and enables reuse of existing content, easy encoding and recording, etc. This clause introduces how to use the 3GPP file format as specified in TS 26.244 [4] for DASH segment formats.

9.2 Segment Types and Formats

9.2.1 Segment Types

3GP-DASH defines a Segment format that is used in the delivery of media data over HTTP. A Segment shall contain one or more boxes in accordance with the boxed structure of the ISO-base media file format [11].

Three different Segment types are defined for 3GP-DASH.

1) Initialisation Segment which may use the MIME type as defined in Annex A of TS 26.244 [4] .

2) Media Segment which may use the MIME type 'video/vnd.3gpp.segment' as defined in Annex A.1.4 of TS 26.244 [4] if accessed through a URL without byte ranges.

3) Self-Initialising Media Segment which may use the MIME type as defined in Annex A of TS 26.244 [4] .

For 3GP-DASH the following applies:

- In all cases for which a Representation contains more than one Media Segment, the following applies:

  - The Initialisation Segment as defined in clause 9.2.2 shall be present. The Initialisation Segment shall be available for the 3GP-DASH client before any Media Segment is processed within the Representation.

  - Media Segments shall not be self-initialising. The Media Segment format is defined in clause 9.2.3.

- In case a Representation contains only a single Media Segment, then either one of the following two options is used:

  1) An Initialisation Segment as defined in clause 9.2.2 and one Media Segment as defined in clause 9.2.3.

  2) One Self-Initialising Media Segment as defined in clause 9.2.4.

9.2.2 Initialisation Segment Format

The Initialisation Segment is conformant with the 3GPP file format, adaptive streaming profile and shall carry '`3gh9`' as compatibility brand.

The Initialisation Segment consists of the '`ftyp`' box, the '`moov`' box, and optionally the '`pdin`' box. The '`moov`' box shall not contain any samples (i.e. the entry_count in the '`stts`', '`stsc`', and '`stco`' boxes shall be set to 0) and is then very small in size. This reduces the start-up time significantly as the Initialisation 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 Initialisation Segment provides the client with the metadata that describes the media content. The client uses the information in the '`moov`' box to identify the available media components and their characteristics.

The Initialisation Segment shall not contain any 'moof' or 'mdat' boxes.

9.2.3 Media Segment Format

A Media Segment conforming to the Media Segment Format for 3GP DASH shall carry "3gmA" as a compatible brand and is defined as follows:

- Each Media Segment may contain an "styp" box.
- If the Media Segment is the last media segment in the Representation, the 'styp' box may carry "lmsg" as a compatible brand.
- 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 and the flag "default-base-is-moof" shall also be set. Absolute byte-offsets shall not be used. In a movie fragment, the durations 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 may contain a "tfdt" box.
- The track fragment adjustment box "tfad" as defined in 3GPP TS26.244 [4] may also be present to maintain compatibility with earlier releases of this specification; care should be taken that the alignment established by the "tfdt" and the time-shifting implied by the "tfad" not be both applied, 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 subsegment documented by the first Segment Index ("sidx") box shall be the entire segment, i.e. the entire segment shall be document by the first Segment Index ("sidx") box.
- A media segment may contain a Subsegment Index box ("ssix"). If present it shall follow immediately after the "sidx" box that documents the same subsegment. This immediately preceding "sidx" shall only index subsegments.
- Further rules on media segments in combination with certain MPD attributes are provided in clause 9.4.

9.2.4 Self-Initialising Media Segment Format

A Self-Initialising Media Segment conforms to the concatenation of an Initialisation Segment as defined in 9.2.2 and a Media Segment as defined in 9.2.3.

9.3 Usage on Server and Client

3GP-DASH uses 3GP files according to the 3GP Adaptive-Streaming profile as specified in TS 26.244 [4]. Content may be prepared as 3GP files according to the 3GP Adaptive-Streaming profile. Initialisation Segments and Media Segments may be generated by segmenting such 3GP files. Segment Index "sidx" boxes may be pre-contained in 3GP files or may be generated during the segmentation process. Clients may store a concatenation of a received Initialisation Segment and a sequence of Media Segments from the same Representation to create a compliant 3GP file according to the Adaptive Streaming profile without accessing any media samples.

NOTE: As specified in TS 26.244, the MPD may be linked or embedded in the "meta" box of the "moov" box. This enables clients to access the MPD from a 3GP file that was made available from other means than 3GP-DASH (e.g. progressive download).

9.4 Media Presentation Authoring Rules for specific MPD flags

9.4.1 General

The content, especially the segments across Representations at the same media time may have been prepared in a joint or at least coordinated manner. To expose these properties to the client, certain flags in the MPD can be set to true to indicate such coordinated content preparation. Clients consuming 3GP-DASH formatted media presentations may benefit from properly authored content when switching between or presenting Representations.

The content authoring rules for the media segments when certain flags are set to true or false are provided in the remainder of this clause.

9.4.2 Segment Alignment

If the `@segmentAlignmentFlag` is set to 'true' it indicates that all presentation start and end times of media components of any particular media type are temporally aligned in all Segments across all Representations with the same value of the `@duration` attribute on Representation level in this Period.

9.4.3 Bitstream Switching

If the `@bitstreamSwitchingFlag` in a **`Period`** or in an **`AdaptationSet`** element is set to "true", all following conditions shall be satisfied:

- The `@segmentAlignmentFlag` for **`Period`** or in an **`AdaptationSet`** element shall be set to "true" and the value of the `@duration` attribute on Representation level is identical for all Representations for which bitstream switching can be applied.
- For any particular media type, all track fragments for all Representations for which bitstream switching can be applied have the same value of track_ID.
- Let X be the concatenation of an Initialization Segment with consecutive Media Segments of a single Representation within a Period, starting with the first Media Segment, and Y be the concatenation wherein the Media Segments with the same constraints come from at least two Representations within the same group. The following applies to all possible instances of X and Y: for any media sample commonly present in X and Y, the decoding time and composition time derived when playing X are identical to the decoding time and composition time, respectively, derived when playing Y, by a legacy player.

9.4.4 Sub-Representation

If a **`SubRepresentation`** element is present in a Representation in the MPD and the **`SubRepresentation`**`@level` is present, then the media segments in this Representation shall include a Segment Index ("`sidx`") box and the Initialisation Segment shall contain the Level Assignment ("`leva`") box.

The attribute `@level` specifies the level to which the described Sub-Representation is associated in the Subsegment Index. Level *n* corresponds to the *n*-th level 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 higher value for `@level` provides an enhancement compared to any lower value of `@level`.

For temporal level assignment, the sample grouping `'tele'` as defined in clause G.4, shall be used.

10 QoE for Progressive Download and DASH

10.1 General

A progressive download or 3GP-DASH client supporting Quality of Experience (QoE) shall report QoE metrics according to the QoE configuration. QoE reporting is optional, but if a 3GP-DASH client reports DASH metrics, it shall report all requested metrics.

The quality metrics are defined in subclause 10.2.

The quality metrics applicable for progressive download are specified in section 10.3. In this case the activation and configuration of QoE reporting framework is achieved by a corresponding OMA DM QoE Management Object as specified in Annex F.

The quality metrics for DASH are specified in section 10.4. In this case, QoE reporting may be triggered using the MPD (i.e. when the **`QualityMetrics`** element is present in the MPD) or using OMA DM QoE Management Object as specified in Annex F. When QoE reporting is triggered via the MPD or OMA DM QoE Management Object, the 3GP-DASH client is expected to collect quality metrics according to the QoE configuration. When using the MPD, the Quality Reporting scheme as defined in section 10.5 may be used.

The quality metric reporting protocol is defined in subclause 10.6. This protocol shall be used when QoE reporting is triggered via the MPD or OMA DM QoE Management Object.

10.2 QoE Metric Definitions

10.2.1 Introduction

This section provides the general QoE metric definitions and measurement framework.

The semantics are defined using an abstract syntax. Section 10.6 provides a mapping to an XML schema. 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`: a 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) and *media time*.

10.2.2 HTTP Request/Response Transactions

Table 25 contains the metric defining the List of HTTP Request/Response Transactions.

Table 25: 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 - MPD delta file - XLink expansion - Initialization Segment - Index Segment - Media Segment
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 index information 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 reported 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 reported 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, MPDDeltaFile, XLinkExpansion, InitialisationSegment, MediaSegment, or IndexSegment. 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 reported metrics.

10.2.3 Representation Switch Events

Table 26 defines the metric to report a list of representation switch events.

Table 26: 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 time of the earliest media sample (out of all media components) played out from the 'to' representation.
		To	String	Representation id identifying the switch-to representation.

10.2.4 Average Throughput

This metric in Table 27 indicates the average throughput that is observed by the client during the measurement interval.

Table 27: Average Throughput

Key		Type	Description
AvgThroughput		Object	Average throughput that is observed by the client during the measurement interval
	Numbytes	Integer	The total number of the content bytes, i.e. the total number of bytes in the body of the HTTP responses, received during the measurement interval.
	Activitytime	Integer	The activity time during the measurement interval in milliseconds. The activity time during the measurement interval is the time during which at least one GET request is still not completed (i.e. excluding inactivity time during the measurement interval).
	T	Real Time	The real time of the start of the measurement interval
	Duration	Integer	The time in milliseconds of the measurement interval
	Accessbearer	String	Access bearer for the TCP connection for which the average throughput is reported
	Inactivitytype	Enum	Type of the inactivity, if known and consistent throughout the reporting period: User request (e.g. pause) Client measure to control the buffer Error case

If the client requests the media segments from the server separately over multiple non-competing parallel TCP connections established over separate access network bearers named as `Accessbearer`, then the average throughput values should be reported as a list of events with average throughput for each access network and associated access network bearer information reported separately, following the same guidelines as described above.

10.2.5 Initial Playout Delay

This metric in Table 28 signals the initial playout delay at the start of the streaming of the presentation.

Table 28: Initial Playout Delay

Key	Type	Description
InitialPlayoutDelay	unsignedInt	The initial playout delay is measured as the time in milliseconds from the fetch of the first media segment (or sub-segment) and the time at which media is retrieved from the client buffer.

10.2.6 Buffer Level

Table 29 defines the metric to report a list of buffer level status events.

Table 29: 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.

10.2.7 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. +/-10 ms).

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 30 defines the play list event metric.

Table 30: 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) - 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.
Traceentry	Objects	Single entry in the list.
representationid	String	The id of the representation from which the samples were taken. This is an optional parameter and should not be reported in case of progressive download.
subreplevel	unsignedInt	If not present, this metrics concerns the representation as a whole. If present, subreplevel indicates the greatest subrepresentation level 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.

10.2.8 MPD Information

This metric can be used to report Representation information from the MPD, so that reporting servers without direct access to the MPD can understand the used media characteristics.

The metric is reported whenever the client sends any other quality metrics report containing references to a Representation which MPD information has still not been reported.

Table 31 defines the MPD information for quality reporting.

Table 31: MPD Information for Quality Reporting

Key		Type	Description
MPDInformation		Object	
	Representationid	String	@id of the representation reported in this metrics
	Subreplevel	unsignedInt	If present, @level of the subrepresentation reported in this metrics. If not present, this metrics concerns the representation as a whole.
	Mpdinfo	RepresentationType	Provides the MPD information for the representation or subrepresentation identified by representationid and subreplevel, if present. The following attributes and elements shall be present within mpdinfo if they are present for the identified representation or subrepresentation and their values shall be identical to those presented in the MPD: @bandwidth, @qualityRanking, @width, @height, @mimeType, and @codecs.

10.3 Quality Metrics for Progressive Download

The following metrics shall be supported by progressive download clients supporting the QoE reporting feature:

- List of HTTP Request/Response Transactions (Section 10.2.2),
- Average Throughput (Section 10.2.4),
- Initial Playout Delay (Section 10.2.5),
- Buffer Level (Section 10.2.6),
- Play List (Section 10.2.7).

10.4 Quality Metrics for DASH

The following metrics shall be supported by 3GP-DASH clients supporting the QoE reporting feature:

- List of HTTP Request/Response Transactions (Section 10.2.2),
- List of of Representation Switch Events (Section 10.2.3),
- Average Throughput (Section 10.2.4),
- Initial Playout Delay (Section 10.2.5),
- Buffer Level (Section 10.2.6),
- Play List (Section 10.2.7), and
- MPD Information (Section 10.2.8).

The @metrics attribute contains a list of QM metric keys listing all metrics that the DASH shall collect and report.

The semantics of the attributes within the **QualityMetrics** element are provided in Table 32. The XML-syntax of a **QualityMetrics** element is provided in Table 33.

Table 32: Semantics of QualityMetrics element

Element or Attribute Name	Use	Description
QualityMetrics		
@metrics	M	This attribute lists all quality metrics (as a list of QM keys as defined in section 10.2, separated by a whitespace) that the client shall report. Certain keys allow specifying a measurement interval or

Element or Attribute Name	Use	Description
		period over which a single value of the metric is derived and potentially also other parameters controlling the collection of the metrics. The parameters, if any, are included in parenthesis after the key and their semantics are specified in clause 10.2 with the metric definition itself.
Range	0..N	When specified, it indicates the time period during which QM collection is requested. When not present, QM collection is requested for the whole duration of the content.
@starttime	O	When specified, it indicates the start time of the QM collection operation. When not present, QM collection is requested from the beginning of content consumption. For Live sessions, the start time is indicated in wallclock time. For OnDemand sessions, the start time is indicated in media time.
@duration	O	When specified, it indicates the duration of the QM collection interval. For Live sessions, the duration is indicated in wallclock time. For OnDemand session, the range is indicated in media time.
QualityReporting	0...N	Descriptor that provides information about the requested Quality Reporting method and formats. See clause 10.6 for the 3GP-DASH quality reporting schemes.
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 33: XML-Syntax of `QualityMetrics` element

```
<!-- QoE Collection and Reporting -->
<xs:complexType name="QualityMetricsType">
    <xs:sequence>
        <xs:element name="QualityReporting" 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>

<xs:complexType name="RangeType">
    <xs:sequence>
        <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="startTime" type="xs:unsignedInt" use="optional"/>
    <xs:attribute name="duration" type="xs:duration" use="required"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

10.5 Quality Reporting Scheme for DASH

This section specifies a 3GP-DASH quality reporting scheme.

The quality reporting scheme is signaled using in the **`QualityReporting`** element in the **`QualityMetrics`** element. The URN to be used for the **`QualityReporting`**`@schemeIdUri` shall be "`urn:3GPP:ns:PSS:DASH:QM10`".

The reporting scheme shall use the quality reporting protocol defined in section 10.6.

The semantics and XML syntax of the scheme information for the 3GP-DASH quality reporting scheme are specified in Table 34 and Table 35, respectively.

Table 34: Semantics of Quality Reporting Scheme Information

Element or Attribute Name	Use	Description
@apn	O	This attribute gives the access point that should be used for sending the QoE reports.
@format	O	This field gives the requested format for the reports. Possible formats are: 'uncompressed' and 'gzip'.
@samplepercentage	O	Percentage of the clients that should report QoE. The client uses a random number generator with the given percentage to find out if the client should report or not.
@reportingserver	M	The reporting server URL to which the reports will be sent.
@reportinginterval	O	Indicates the time(s) reports should be sent. If not present, then the client should send a report after the streaming session has ended. If present, @reportingInterval=n indicates that the client should send a report every n-th second provided that new metrics information has become available since the previous report.
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 35: Syntax of Quality Reporting Scheme Information

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="urn:3GPP:ns:PSS:AdaptiveHTTPStreaming:2009:qm"
    attributeFormDefault="unqualified"
    elementFormDefault="qualified"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreaming:2009:qm">

    <xs:annotation>
        <xs:appinfo>3GPP DASH Quality Reporting</xs:appinfo>
        <xs:documentation xml:lang="en">
            This Schema defines the quality reporting scheme information for 3GPP DASH.
        </xs:documentation>
    </xs:annotation>

    <xs:element name="ThreeGPQualityReporting" type="SimpleQualityReportingType"/>

    <xs:complexType name="SimpleQualityReportingType">
        <xs:attribute name="apn" type="xs:string" use="optional"/>
        <xs:attribute name="format" type="FormatType" use="optional"/>
        <xs:attribute name="samplePercentage" type="xs:double" use="optional"/>
        <xs:attribute name="reportingServer" type="xs:anyURI" use="required"/>
        <xs:attribute name="reportingInterval" type="xs:unsignedInt" use="optional"/>
    </xs:complexType>

    <xs:simpleType name="FormatType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="uncompressed" />
            <xs:enumeration value="gzip" />
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

10.6 Quality Reporting Protocol

10.6.1 General

The quality reporting protocol consists of:

- The report format defined in section 10.6.2
- The reporting protocol defined in section 10.6.3

10.6.2 Report Format

The QoE report is formatted as an XML document that complies with the following XML schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="urn:3gpp:metadata:2011:HSD:receptionreport"
    xmlns="urn:3gpp:metadata:2011:HSD:receptionreport" elementFormDefault="qualified">

    <xs:element name="ReceptionReport" type="ReceptionReportType"/>

    <xs:complexType name="ReceptionReportType">
        <xs:choice>
            <xs:element name="QoeReport" type="QoeReportType" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:any namespace="##other" processContents="skip" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:choice>
        <xs:attribute name="contentURI" type="xs:anyURI" use="required"/>
        <xs:attribute name="clientID" type="xs:string" use="optional"/>
    </xs:complexType>

    <xs:complexType name="QoeReportType">
        <xs:sequence>
            <xs:element name="QoeMetric" type="QoeMetricType" minOccurs="1"
maxOccurs="unbounded"/>
            <xs:any namespace="##other" processContents="skip" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="periodID" type="xs:string" use="required"/>
        <xs:attribute name="reportTime" type="xs:dateTime" use="required"/>
        <xs:attribute name="reportPeriod" type="xs:unsignedInt" use="required"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>
    <xs:complexType name="QoeMetricType">
        <xs:choice>
            <xs:element name="HttpList" type="HttpListType"/>
            <xs:element name="RepSwitchList" type="RepSwitchListType"/>
            <xs:element name="AvgThroughput" type="AvgThroughputType" maxOccurs="unbounded"/>
            <xs:element name="InitialPlayoutDelay" type="xs:unsignedInt"/>
            <xs:element name="BufferLevel" type="BufferLevelType"/>
            <xs:element name="PlayList" type="PlayListType"/>
            <xs:element name="MPDInformation" type="MpdInformationType" maxOccurs="unbounded"/>
        </xs:choice>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:complexType name="HttpListType">
        <xs:choice>
            <xs:element name="HttpListEntry" type="HttpListEntryType" maxOccurs="unbounded"/>
        </xs:choice>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:complexType name="HttpListEntryType">
        <xs:choice>
            <xs:element name="Trace" type="HttpThroughputTraceType" maxOccurs="unbounded"/>
        </xs:choice>
        <xs:attribute name="tcpid" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="type" type="ExtensibleHttpEntryResourceType" use="optional"/>
        <xs:attribute name="url" type="xs:string" use="required"/>
        <xs:attribute name="actualUrl" type="xs:string" use="optional"/>
        <xs:attribute name="range" type="xs:string" use="optional"/>
        <xs:attribute name="trequest" type="xs:dateTime" use="required"/>
        <xs:attribute name="tresponse" type="xs:dateTime" use="required"/>
        <xs:attribute name="responsecode" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="interval" type="xs:unsignedInt" use="optional"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:simpleType name="HttpEntryResourceType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="MPD"/>
            <xs:enumeration value="MPDDeltaFile"/>
            <xs:enumeration value="XLinkExpansion"/>
```

```
            <xs:enumeration value="InitialisationSegment"/>
            <xs:enumeration value="IndexSegment"/>
            <xs:enumeration value="MediaSegment"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="StringPatternType">
        <xs:restriction base="xs:string">
            <xs:pattern value="x:\S.*"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="ExtensibleHttpEntryResourceType">
        <xs:union memberTypes="HttpEntryResourceType StringPatternType"/>
    </xs:simpleType>

    <xs:complexType name="HttpThroughputTraceType">
        <xs:attribute name="s" type="xs:dateTime" use="required"/>
        <xs:attribute name="d" type="xs:unsignedInt" use="required"/>
        <xs:attribute name="b" type="xs:unsignedInt" use="required"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:complexType name="RepSwitchListType">
        <xs:choice>
            <xs:element name="RepSwitchEvent" type="RepSwitchEventType" maxOccurs="unbounded"/>
        </xs:choice>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:complexType name="RepSwitchEventType">
        <xs:attribute name="to" type="xs:string" use="required"/>
        <xs:attribute name="mt" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="t" type="xs:dateTime" use="optional"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:complexType name="AvgThroughputType">
        <xs:attribute name="numBytes" type="xs:unsignedInt" use="required"/>
        <xs:attribute name="activityTime" type="xs:unsignedInt" use="required"/>
        <xs:attribute name="t" type="xs:dateTime" use="required"/>
        <xs:attribute name="duration" type="xs:unsignedInt" use="required"/>
        <xs:attribute name="accessbearer" type="xs:string" use="optional"/>
        <xs:attribute name="inactivityType" type="InactivityType" use="optional"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:simpleType name="InactivityType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Pause"/>
            <xs:enumeration value="BufferControl"/>
            <xs:enumeration value="Error"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:complexType name="BufferLevelType">
        <xs:choice>
            <xs:element name="BufferLevelEntry" type="BufferLevelEntryType"
maxOccurs="unbounded"/>
        </xs:choice>

        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:complexType name="BufferLevelEntryType">
        <xs:attribute name="t" type="xs:dateTime" use="required"/>
        <xs:attribute name="level" type="xs:unsignedInt" use="required"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:complexType name="PlayListType">
        <xs:choice>
            <xs:element name="Trace" type="PlayListEntryType" maxOccurs="unbounded"/>
        </xs:choice>

        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:complexType name="PlayListEntryType">
```

```
        <xs:choice>
            <xs:element name="TraceEntry" type="PlayListTraceEntryType" maxOccurs="unbounded"/>
        </xs:choice>
        <xs:attribute name="start" type="xs:dateTime" use="required"/>
        <xs:attribute name="mstart" type="xs:unsignedInt" use="required"/>
        <xs:attribute name="startType" type="StartType" use="required"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:complexType name="PlayListTraceEntryType">
        <xs:attribute name="representationId" type="xs:string" use="optional"/>
        <xs:attribute name="subrepLevel" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="start" type="xs:dateTime" use="required"/>
        <xs:attribute name="mstart" type="xs:unsignedInt" use="required"/>
        <xs:attribute name="duration" type="xs:unsignedInt" use="required"/>
        <xs:attribute name="playbackSpeed" type="xs:double" use="optional"/>
        <xs:attribute name="stopReason" type="StopReasonType" use="optional"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:simpleType name="StartType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="NewPlayoutRequst"/>
            <xs:enumeration value="Resume"/>
            <xs:enumeration value="OtherUserRequest"/>
            <xs:enumeration value="StartOfMetricsCollectionPeriod"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="StopReasonType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="RepresentationSwitch"/>
            <xs:enumeration value="Rebuffering"/>
            <xs:enumeration value="UserRequest"/>
            <xs:enumeration value="EndOfPeriod"/>
            <xs:enumeration value="EndOfContent"/>
            <xs:enumeration value="EndOfMetricsCollectionPeriod"/>
            <xs:enumeration value="Failure"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="MpdInformationType">
        <xs:choice>
            <xs:element name="Mpdinfo" type="RepresentationType" maxOccurs="unbounded"/>
        </xs:choice>
        <xs:attribute name="representationId" type="xs:string" use="required"/>
        <xs:attribute name="subrepLevel" type="xs:unsignedInt" use="optional"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:complexType name="RepresentationType">
        <xs:attribute name="codecs" type="xs:string" use="required"/>
        <xs:attribute name="bandwidth" type="xs:unsignedInt" use="required"/>
        <xs:attribute name="qualityRanking" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="frameRate" type="xs:double" use="optional"/>
        <xs:attribute name="width" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="height" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="mimeType" type="xs:string" use="required"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>

    <xs:simpleType name="DoubleVectorType">
        <xs:list itemType="xs:double"/>
    </xs:simpleType>

    <xs:simpleType name="StringVectorType">
        <xs:list itemType="xs:string"/>
    </xs:simpleType>

</xs:schema>
```

10.6.3 Reporting Protocols

If a specific metrics server has been configured, the client shall send QoE reports using the HTTP (RFC 2616) [9] POST request carrying XML formatted metadata in its body.

An example QoE reporting based on HTTP POST request signalling is shown below:

```
POST http://www.exampleserver.com HTTP/1.1
Host: 192.68.1.1
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0)
Content-Type: text/xml; charset=utf-8
Content-Length: 4408

<?xml version="1.0" encoding="UTF-8"?>
<receptionReport ContentURI="http://www.example.com/content/content.mpd" ClientID="35848574673"
xmlns="urn:3gpp:metadata:2011:HSD:receptionreport">
    <qoeReport PeriodID="Period1" ReportTime="2011-02-16T09:00:00" ReportPeriod="500">
        <qoeMetric>
            <HttpList>
                <HttpListEntry type="MPD" url="http://www.example.com/content/content.mpd"
trequest="2011-02-16T08:59:30" tresponse="2011-02-16T08:59:31" />
                <HttpListEntry type="InitialisationSegment"
url="http://www.example.com/content/initRep1.3gp" trequest="2011-02-16T08:59:40"
tresponse="2011-02-16T08:59:41" />
                <HttpListEntry type="InitialisationSegment"
url="http://www.example.com/content/initRep2.3gp" trequest="2011-02-16T08:59:41"
tresponse="2011-02-16T08:59:42" />
                <HttpListEntry type="InitialisationSegment"
url="http://www.example.com/content/initRep3.3gp" trequest="2011-02-16T08:59:42"
tresponse="2011-02-16T08:59:43" />
            </HttpList>
        </qoeMetric>
        <qoeMetric>
            <InitialPlayoutDelay>10000</InitialPlayoutDelay>
        </qoeMetric>
    </qoeReport>
    <qoeReport PeriodID="Period1" ReportTime="2011-02-16T09:08:20" ReportPeriod="500">
        <qoeMetric>
            <BufferLevel>
                <BufferLevelEntry t="2011-02-16T09:08:19" level="84673"/>
                <BufferLevelEntry t="2011-02-16T09:08:20" level="93874"/>
            </BufferLevel>
        </qoeMetric>
        <qoeMetric>
            <RepSwitchList>
                <RepSwitchEvent to="Rep2"/>
                <RepSwitchEvent to="Rep3"/>
            </RepSwitchList>
        </qoeMetric>
    </qoeReport>
</receptionReport>
```

Annex A (informative): Client Behaviour

A.1 Introduction

The information on client behaviour is purely informative and does not imply any normative procedures on client implementations.

A.2 Overview

A 3GP-DASH client is guided by the information provided in the MPD. It is assumed that the client has access to the MPD that it fetched at time *FetchTime* at its initial location if no Location element is present, or at a location specified in any present **MPD.Location** element. *FetchTime* is defined at the client as the time the client has succeeded in fetching an instance of an MPD. It is considered as a successful fetching either if the client obtains an updated MPD or the client verifies that the MPD has not been updated since the previous fetching. An example client behaviour is introduced. For providing a continuous streaming service to the user:

1) The client parses the MPD and 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 specified in clause A.3.

2) The client selects one or multiple Representations based on the information in the Representation attributes and other information, e.g. available bandwidth and client capabilities. Representations assigned to group 0 are presented without any other Representation. Representations assigned to a non-zero group are typically presented in combination with Representations from groups other than their own, not including the group 0. For each selected Representation, the client acquires the Initialisation Segment, if present, and the Media Segments.

3) The client accesses the content by requesting Segments or byte ranges of Segments. The client requests the Media Segments of the selected Representations by using the generated Segment list.

4) The client buffers media of at least `@minBufferTime` duration before starting the presentation. Then, once identified a Representation Access Point (RAP), it starts rendering (in wall-clock-time) of this RAP not before **`MPD`**`@availabilityStartTime` + *PeriodStart* + T_{RAP} and not after **`MPD`**`@availabilityStartTime` + *PeriodStart* + T_{RAP} + `@timeShiftBufferDepth`.

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 available bandwidth. With any request for a Media Segment containing a representation access point, the client may switch to a different Representation.

6) When moving forward, i.e. the *NOW* time advancing, the client consumes the available segments. With each advance in *NOW* the client possibly expands the list of available Segments for each Representation according to the procedures specified in clause A.3. If:

   a) the `@mediaPresentationDuration` attribute is not declared or if any media described in the MPD does not reach to the end of the Media Presentation, and

   b) 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 on Representation level) of the media described in the MPD for any consuming or to be consumed Representation.

   then the client should fetch a new MPD, with new fetch time *FetchTime*. Once received the client now takes into account the possibly updated MPD and the new *FetchTime* in the generation of the accessible Segment Lists.

In the following 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 3GP-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 synchronised with a DASH server, which is in turn is expected to be synchronised 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, 3GP-DASH clients are expected to synchronize their clocks to a globally accurate time standard.

Table A.1: Segment List

Parameter Name	Cardinality	Description
Segments	1	Provides the Segment URL list.
InitialisationSegment	0, 1	Describes the Initialisation Segment. If not present each Media Segment is self-initialising.
URL	1	The URL where to access the Initialisation Segment (the client would restrict the URL with a byte range if one is provided in the MPD).
MediaSegment	1 … N	Describes the accessible Media Segments.
startTime	1	The approximate start time of the Media Segment in the Period relative to the start time of Period. To obtain the start time of the Media Segment in the Media Presentation, the start time of the Period needs to be added for On-Demand services. This value can also be used to determine the segment availability start time.
duration	1	The duration for the segment
URL	1	The URL where to access the Media Segment (the client would restrict the URL with a byte range if one is provided in the MPD).

From the information in the MPD a derived **`SegmentInfo`** element as described in clauses 8.4.4.1, 8.4.4.2 and 8.4.4.3 may be derived. Each derived **`SegmentInfo`** element is used to generate a list of accessible Segments for each Representation.

The following rules apply for a derived **`SegmentInfo`** element:

- If the derived **`SegmentInfo`** element contains a **`URLtemplate`** element, then the procedures in clause A.3.2 are used to generate a list of Media Segments.
- If the derived **`SegmentInfo`** element contains one or more **`Url`** elements, possibly contained in one or more **`SegmentList`** elements, providing a set of explicit URL(s) for Media Segments, then the procedures in clause A.3.3 are used to generate a list of Media Segments.
- If the **`MPD`**`@type` is '`Live`', then the restrictions on Media Segment Lists as provided in clause A.3.4 are applied.

The client should only request Segments that are included in the Segment List at time instant *NOW*, i.e. segment that are available at this time instant.

A.3.2 Template-based Generation of Media Segment List

If the derived **`SegmentInfo`** element contains a **`URLtemplate`** element, then the procedures in this clause are used to generate a list of Media Segment parameters, i.e. segment URLs and start times. No byte ranges are associated with the URLs.

Assume that the Period end time documented in the current MPD with fetch time *FetchTime* is defined as *PeriodEndTime*. For any Period in the MPD except for the last one, the *PeriodEndTime* is obtained as the value of the sum of **MPD**@availabilityStartTime and *PeriodStart* of the next Period. For the last Period in the MPD:

- if the @mediaPresentationDuration attribute is present, then *PeriodEndTime* is defined as the end time of the Media Presentation.
- if the @mediaPresentationDuration attribute is not present, then *PeriodEndTime* is defined as *FetchTime* + @minimumUpdatePeriodMPD).

If the derived **SegmentInfo** Element contains a **UrlTemplate** element containing a @sourceURL attribute, then this **UrlTemplate** is used as the valid **UrlTemplate** for this Representation. Otherwise, the @sourceUrlTemplate attribute is present; in this case the *$RepresentationID$* identifier in the @sourceUrlTemplate attribute is replaced by the value of the @id attribute on Representation level and the result string is used as the @sourceURL attribute in the **UrlTemplate** element that is valid for the current Representation.

Assume that Media Segments within a Representation have been assigned consecutive indices *i*=1,2,3…., i.e. the first Media Segment has been assigned the index *i*=1, the second Media Segment has been assigned the index *i*=2, and so on.

A valid list of Media Segments with Segment indices *i*, MediaSegment.StartTime[*i*] and MediaSegment.URL[*i*], *i*=@startIndex, @startIndex+1, …, is obtained as follows using the @duration attribute for this Representation:

1) Set *i*=@startIndex.
2) The start time of the first Media Segment is obtained as (@startIndex -1)* @duration, i.e. MediaSegment.StartTime[1] = (@startIndex-1)*@duration. If the @duration attribute is not provided, then the MediaSegment.StartTime[1] of the only provided Segment is set to 0.
3) The URL of the Media Segment *i*, MediaSegment.URL[*i*], is obtained by replacing the $*Index$* identifier by *i* in the @sourceURL string of the valid **UrlTemplate**..
4) If ((**MPD**@availabilityStartTime + *PeriodStart* + MediaSegment.StartTime[*i*] + @duration) < *PeriodEndTime*)
   - then
     - A new Media Segment is added to the list, i.e. *i* = *i* + 1;
     - The duration is set to MediaSegment.duration[*i*] = @duration
     - MediaSegment.StartTime[*i*] = MediaSegment.StartTime[*i*-1] + @duration.
     - Proceed with step 3.
   - else
     - A new Media Segment is added to the list, i.e. *i* = *i* + 1;
     - MediaSegment.StartTime[*i*] = MediaSegment.StartTime[*i*-1] + @duration.
     - The guaranteed duration is set to MediaSegment.duration[*i*] = *PeriodEndTime* - MediaSegment.StartTime[*i*]
     - The restrictions as specified in clause A.3.4 are applied for the creation of the accessible list of Media Segments.

A.3.3 Playlist-based Generation of Media Segment List

If the derived **SegmentInfo** element contains one or more **Url** elements, possibly contained in one or more **SegmentList** elements providing a set of explicit URL(s) for Media Segments, , then the procedures specified in this clause apply to generate a valid list of accessible Media Segment URLs and start times described in each derived **SegmentInfo** element.

Assume that Media Segments within a Representation have been assigned consecutive indices *i*=1,2,3…., i.e. the first Media Segment has been assigned *i*=1, the second Media Segment has been assigned *i*=2, and so on.

A valid list of Media Segments with segment indices *i*=`@startIndex`, `@startIndex` +1, …, MediaSegment.StartTime[*i*] and MediaSegment.URL[*i*] is obtained as follows:

1) Set *i*=`@startIndex`.

2) The URL of the Media Segment *i*, MediaSegment.URL[*i*], is obtained as the `@sourceURL` attribute of the (*i*-`@startIndex` +1)-th **Url** element in the derived **`SegmentInfo`** element taking into account URI reference resolution, restricted to the byte range specified in the `@range` attribute of the same **Url** element, if present.

3) If the `@duration` attribute is provided, then the MediaSegment.StartTime[*i*] of Media Segment *i* is obtained as (*i*-1)*`@duration`. If the `@duration` attribute is not provided, then the MediaSegment.StartTime[1] of the only provided Segment is set to 0.

4) If this is not the last **Url** element, a new Media Segment is added to the list, i.e. *i* = *i* + 1, and proceed with step 2; Otherwise proceed with step 5.

5) The restrictions as specified in clause A.3.4 are applied for the creation of the accessible list of Media Segments.

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*.

Generally, Segments are only available for any time *NOW* between `@availabilityStartTime` and `@availabilityEndTime`. For times *NOW* outside this window, no Segments are available.

In addition, for live services, assume the variable *CheckTime* associated to an the MPD with *FetchTime* is defined as:

- If the `@minimumUpdatePeriodMPD` attribute is provided, then the check time is defined as the sum of the fetch time of this operating MPD and the value of this attribute, i.e. *CheckTime* = *FetchTime* + `@minimumUpdatePeriodMPD`.

- If the `@minimumUpdatePeriodMPD` attribute is not provided, external means are used to determine *CheckTime*, such as a priori knowledge, or HTTP cache headers, etc.

The *CheckTime* is defined on the MPD-documented media time axis; when the client"s playback time reaches or gets close to *CheckTime* - **`MPD`**`@minBufferTime` it should fetch a new MPD.

Then, the Media Segment list is further restricted by the *CheckTime* together with the MPD attribute **`MPD`**`@timeShiftBufferDepth` such that only Media Segments for which the sum of the start time of the Media Segment and the Period start time falls in the interval [*NOW* - **`MPD`**`@timeShiftBufferDepth` - **`SegmentInfo`**`@duration`, min(*CheckTime, NOW*)] are included.

A.4 Seeking

Assume that a client attempts to seek to a specific presentation time *tp* in a Representation with start time *PeriodStart*. *PeriodStart* defines the absolute start time of the Media Segment in the Media Presentation, i.e. for On-Demand services the start time of the Period needs to be added and for live services, in addition also the value of the **`MPD`**`@availabilityStartTime` attribute needs to be added. Before accessing the Media Segments of a Representation, the client needs to download the Initialisation Segment, if present.

Based on the MPD, the client has access to the Media Segment start time and Media Segment URL of each Segment in the Representation. The Segment index *segment_index* of the Segment most likely to contain media samples for presentation time *tp* is obtained as the maximum Segment index *i*, for which the start time MediaSegment[*i*].StartTime is smaller or equal to the presentation time relative to the Representation start time *tp-PeriodStart*, i.e.

segment_index = max { *i* | MediaSegment[*i*].StartTime <= *tp* - *PeriodStart* }.

The Segment URL is obtained as MediaSegment[*segment_index*].URL.

Note that timing information in the MPD may be approximate due to issues related to placement of Representation 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 *tp* and the media data for presentation time *tp* 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 file, or the preceeding file 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 *tp* and the start of the retrieved Segment is always available.

For accurate seeking to a presentation time *tp*, the 3GP-DASH Client needs to access a representation access point (RAP). To determine the representation access point in a Media Segment in case of 3GP-DASH, the client may, for example, use the information in the "sidx" box if present to locate the representation access points and the corresponding presentation time in the Media Presentation. In the case that a Segment is a 3GPP movie fragment, it is also possible for the client to use information within the "moof" and "mdat" boxes, for example, to locate RAPs and obtain the necessary presentation time from the information in the movie fragment and the segment start time derived from the MPD. If no RAP with presentation time before the requested presentation time *tp* is available, the client may either access the previous Segment or may just use the first representation access point as the seek result. When Media Segments start with a RAP, these procedures are simple.

Also note that not necessarily all information of the Media Segment needs to be downloaded to access the presentation time *tp*. The client may for example initially request the "sidx" box from the beginning of the Media Segment using byte range requests. By use of the "sidx", segment timing can be mapped to byte ranges of the Segment. By continuously using partial HTTP 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 fragment after the last requested fragment.

If a specific **`Representation`** or **`SubRepresentation`** element includes the `@maxPlayoutRate` attribute, then this Representation or Sub-representation may be used for the fast-forward trick mode. If a specific **`Representation`** or **`SubRepresentation`** element includes the `@codingDependency` attribute with value set to `'false'`, then this Representation or Sub-representation may be used for both fast-forward and fast-rewind trick modes. 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.

The client may use multiple Representations to support trick mode behaviour.

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 RAP 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 RAP in the 'new' Representation enables seamless switching.

Aligning RAPs across different Representations may be advantageous in locating RAPs in other Representations.

A.7 Reaction to Error Codes

The HTTP Streaming client provides a streaming service to the user by issuing HTTP requests for Segments at appropriate times. The HTTP Streaming 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 clause as to how an HTTP client may react to such error codes.

If the HTTP Client receives an HTTP client or server error (i.e. messages with 4xx or 5xx error code), the client should respond appropriately to the error code.

If the HTTP Client receives a repeated HTTP error for the request of an MPD, the appropriate response may involve terminating the streaming service.

If the HTTP Client receives an HTTP client error (i.e. messages with 4xx error code) for the request of an Initialisation Segment, the Period containing the Initialisation Segment may not be available anymore or may not be available yet. In this case the client should check if the precision of the time synchronization to a globally accurate time standard is sufficiently accurate. In case of repeated errors, the client should check for an update of the MPD.

If the HTTP 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 this case the client should check if the precision of the time synchronization to a globally accurate time standard is sufficiently accurate. In case of repeated errors, 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. The client may also check for alternative representations 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.

During long live streaming sessions, 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. Clients may avoid these anomalies by using the Producer Reference Time boxes as defined in clause G.5 as follows. The pace r1 of the encoder clock in relation to the UTC is recovered from Producer Reference Time boxes. If the relative pace r1 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 r2 of the receiver playout clock in relation to UTC is created by accessing a UTC source. A timescale multiplication factor c is equal to r1/r2. 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.

Annex B (normative): Media Presentation Description Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009"
    attributeFormDefault="unqualified" elementFormDefault="qualified"
    xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
    <xs:annotation>
        <xs:appinfo>Media Presentation Description</xs:appinfo>
        <xs:documentation xml:lang="en"> This defines 3GPP Media Presentation Description!
        </xs:documentation>
    </xs:annotation>

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

    <!-- 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"/>
            <xs:element name="DeltaSupport" type="DeltaSupportType" minOccurs="0"/>
            <xs:element name="Location" type="MpdUrlType" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="BaseURL" type="BaseUrlType" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="QualityMetrics" type="QualityMetricsType" minOccurs="0"
                maxOccurs="unbounded"/>
            <xs:element name="Period" type="PeriodType" maxOccurs="unbounded"/>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="profiles" type="UriVectorType"/>
        <xs:attribute name="type" type="PresentationType" default="OnDemand"/>
        <xs:attribute name="availabilityStartTime" type="xs:dateTime"/>
        <xs:attribute name="availabilityEndTime" type="xs:dateTime"/>
        <xs:attribute name="mediaPresentationDuration" type="xs:duration"/>
        <xs:attribute name="minimumUpdatePeriodMPD" type="xs:duration"/>
        <xs:attribute name="minBufferTime" type="xs:duration"/>
        <xs:attribute name="timeShiftBufferDepth" type="xs:duration"/>
        <xs:attribute name="suggestedPresentationDelay" type="xs:duration"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- Type for space delimited list of URIs -->
    <xs:simpleType name="UriVectorType">
        <xs:list itemType="xs:anyURI"/>
    </xs:simpleType>

    <!-- Type of presentation - live or on-demand -->
    <xs:simpleType name="PresentationType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="OnDemand"/>
            <xs:enumeration value="Live"/>
        </xs:restriction>
    </xs:simpleType>

    <!-- Supplementary URL to the one given as attribute -->
    <xs:complexType name="BaseUrlType">
        <xs:simpleContent>
            <xs:extension base="xs:anyURI">
                <xs:anyAttribute namespace="##other" processContents="lax"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
    <xs:complexType name="MpdUrlType">
        <xs:simpleContent>
            <xs:extension base="xs:anyURI">
```

```
                <xs:anyAttribute namespace="##other" processContents="lax"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

    <!-- Program information for a presentation -->
    <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="moreInformationURL" type="xs:anyURI"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!--DeltaSupport for the MPD -->
    <xs:complexType name="DeltaSupportType">
        <xs:sequence>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="sourceURL" type="xs:anyURI" use="required"/>
        <xs:attribute name="availabilityDuration" type="xs:duration"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- QoE Collection and Reporting -->
    <xs:complexType name="QualityMetricsType">
        <xs:sequence>
            <xs:element name="QualityReporting" 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>

    <xs:complexType name="RangeType">
        <xs:sequence>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="startTime" type="xs:unsignedInt" use="optional"/>
        <xs:attribute name="duration" type="xs:duration" use="required"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- Period of a presentation -->
    <xs:complexType name="PeriodType">
        <xs:sequence>
            <xs:element name="SegmentInfoDefault" type="SegmentInfoDefaultType"
minOccurs="0"/>
            <xs:element name="Representation" type="RepresentationType" minOccurs="0"
                maxOccurs="unbounded"/>
            <xs:element name="AdaptationSet" type="AdaptationSetType" 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="segmentAlignmentFlag" type="xs:boolean" default="false"/>
        <xs:attribute name="bitStreamSwitchingFlag" type="xs:boolean" default="false"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- Default Segment access information -->
    <xs:complexType name="SegmentInfoDefaultType">
        <xs:sequence>
            <xs:element name="BaseURL" type="BaseUrlType" minOccurs="0"
maxOccurs="unbounded"/>
```

```
            <xs:element name="InitialisationSegmentURL" type="UrlType" minOccurs="0"/>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attributeGroup ref="SegmentInfoAttrGroup"/>
        <xs:attribute name="sourceURLTemplate" type="xs:string"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- grouping attributes common to SegmentInfo and SegmentInfoDefault -->
    <xs:attributeGroup name="SegmentInfoAttrGroup">
        <xs:attribute name="duration" type="xs:duration"/>
        <xs:attribute name="startIndex" type="xs:unsignedInt" default="1"/>
    </xs:attributeGroup>

    <!-- RepresentationBase type; extended by other Representation-related types -->
    <xs:complexType name="RepresentationBaseType" abstract="true">
        <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="ContentProtection" 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:choice>
        <xs:attribute name="width" type="xs:unsignedInt"/>
        <xs:attribute name="height" type="xs:unsignedInt"/>
        <xs:attribute name="frameRate" type="xs:double"/>
        <xs:attribute name="lang" type="LangVectorType"/>
        <xs:attribute name="mimeType" type="xs:string"/>
        <xs:attribute name="codecs" type="xs:string"/>
        <xs:attribute name="group" type="xs:unsignedInt"/>
        <xs:attribute name="startWithRAP" type="xs:boolean"/>
        <xs:attribute name="maxPlayoutRate" type="xs:double"/>
        <xs:attribute name="codingDependency" type="xs:boolean"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- Type for space delimited list of language codes -->
    <xs:simpleType name="LangVectorType">
        <xs:list itemType="xs:language"/>
    </xs:simpleType>

    <!-- Group to contain information common to an adaptation set;
        extends RepresentationBaseType -->
    <xs:complexType name="AdaptationSetType">
        <xs:complexContent>
            <xs:extension base="RepresentationBaseType">
                <xs:sequence>
                    <xs:element name="SegmentInfoDefault" type="SegmentInfoDefaultType"
                        minOccurs="0"/>
                    <xs:element name="Representation" type="RepresentationType" 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="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="xs:double"/>
                <xs:attribute name="maxFrameRate" type="xs:double"/>
                <xs:attribute name="segmentAlignmentFlag" type="xs:boolean"/>
                <xs:attribute name="bitStreamSwitchingFlag" type="xs:boolean"/>
                <xs:anyAttribute namespace="##other" processContents="lax"/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <!-- A Representation of the presentation content for a specific Period -->
    <xs:complexType name="RepresentationType">
```

```
        <xs:complexContent>
            <xs:extension base="RepresentationBaseType">
                <xs:sequence>
                    <xs:element name="SubRepresentation" type="SubRepresentationType"
minOccurs="0"/>
                    <xs:element name="SegmentInfo" type="SegmentInfoType" minOccurs="0"/>
                    <xs:any namespace="##other" processContents="lax" minOccurs="0"
                        maxOccurs="unbounded"/>
                </xs:sequence>
                <xs:attribute name="id" type="xs:string" use="required"/>
                <xs:attribute name="bandwidth" type="xs:unsignedInt" use="required"/>
                <xs:attribute name="qualityRanking" type="xs:unsignedInt"/>
                <xs:anyAttribute namespace="##other" processContents="lax"/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <!-- SubRepresentation of the presentation content for a specific Period;
        extends RepresentationBaseType -->
    <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:anyAttribute namespace="##other" processContents="lax"/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>

    <!-- Type for space delimited list of strings -->
    <xs:simpleType name="UIntVectorType">
        <xs:list itemType="xs:unsignedInt"/>
    </xs:simpleType>

    <!-- Segment access information -->
    <xs:complexType name="SegmentInfoType">
        <xs:sequence>
            <xs:element name="BaseURL" type="BaseUrlType" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="InitialisationSegmentURL" type="UrlType" minOccurs="0"/>
            <xs:choice minOccurs="0">
                <xs:element name="UrlTemplate" type="UrlTemplateType" minOccurs="0"/>
                <xs:sequence>
                    <xs:element name="Url" type="UrlType" maxOccurs="unbounded"/>
                    <xs:any namespace="##other" processContents="lax" minOccurs="0"
                        maxOccurs="unbounded"/>
                </xs:sequence>
                <xs:element name="SegmentList" type="SegmentListType" minOccurs="0"/>
                <xs:any namespace="##other" processContents="lax" minOccurs="0"
                    maxOccurs="unbounded"/>
            </xs:choice>
        </xs:sequence>
        <xs:attributeGroup ref="SegmentInfoAttrGroup"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- A Segment URL  -->
    <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>

    <!-- A URL template  -->
    <xs:complexType name="UrlTemplateType">
        <xs:sequence>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="sourceURL" type="xs:anyURI"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
```

```
    <!-- SegmentList allows xlink in addition to list of URLs -->
    <xs:complexType name="SegmentListType">
        <xs:sequence>
            <xs:element name="Url" type="UrlType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute ref="xlink:href"/>
        <xs:attribute ref="xlink:actuate" default="onRequest"/>
        <xs:attribute name="startIndex" type="xs:unsignedInt"/>
    </xs:complexType>

    <!-- Generic named descriptive information -->
    <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" use="optional"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

</xs:schema>
```

Namespaces may be used to extend functionalities. As with typical XML practice, any elements or attributes not supported by the terminal shall be ignored. Therefore, all extended elements and attributes added to a **`Representation`** in particular shall be such that they can be safely ignored by 3GP-DASH clients.

Example for valid MPDs are provided in Annex D.

Annex C (normative): Descriptor Scheme Definitions

C.1 Introduction

This annex defines descriptors that are defined in this specification. In particular the following descriptors are defined

- Role descriptor scheme in clause C.2.

C.2 Role Descriptor Scheme

For all Representations or Groups for which a **`Role`** element is provided a value according to the `@value` provided in Table C.1 is defined. Note that **`Role`**`@value` shall be assigned to Representations that contain a media type to which this role is associated.

Table C.1: `Role@value` attríbute values"

Media Type	**`Role@`**`value`	Description
text	`caption`	Closed captioning
text	`subtitle`	Sub-title text
video	`main`	main video component which is intended for presentation if no other information is provided
video	`alternate`	alternate video component (see note 2 below)
video	`supplementary`	supplementary audio to a program that is mainly video (see Note 1 below)
audio	`main`	main audio component which is intended for presentation if no other information is provided
audio	`alternate`	alternate audio component (see note 2 below)
audio	`commentary`	audio with commentary (e.g. director"s commentary)
audio	dub	dubbed audio indicating that it is not the original language
audio	`supplementary`	supplementary video to a program that is mainly audio (see Note 1 below)

NOTE 1: A normal audio/video program labels both the primary audio and video as "main". However, when the two media 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.

NOTE 2: Alternate content should carry other descriptors to indicate in what way it differs from the main content (e.g. viewpoint).

Annex D (informative): MPD Examples

D.1 On-Demand Service

Table D.1 provides an example MPD for an On-Demand service.

Table D.1: Example MPD for an On-Demand Service

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
    type="OnDemand"
    minBufferTime="PT10S"
    mediaPresentationDuration="PT2H"
    availabilityStartTime="2010-04-01T09:30:47Z"
    availabilityEndTime="2010-04-07T09:30:47Z"
    timeShiftBufferDepth="PT30M"
    xsi:schemaLocation="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009 3GPP-Rel10-MPD.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
    <ProgramInformation moreInformationURL="http://www.example.com">
        <Title>Example</Title>
    </ProgramInformation>
    <BaseURL>http://www.example.com</BaseURL>
    <Period start="PT0S">
        <Representation
            mimeType="video/3gpp; codecs=s263, samr"
            bandwidth="256000"
            id="256">
            <SegmentInfo duration="PT10S">
                <BaseURL>"rep1"</BaseURL>
                <InitialisationSegmentURL sourceURL="seg-init.3gp"/>
                <Url sourceURL="seg-1.3gp"/>
                <Url sourceURL="seg-2.3gp"/>
                <Url sourceURL="seg-3.3gp"/>
            </SegmentInfo>
        </Representation>
        <Representation
            mimeType="video/3gpp; codecs=mp4v.20.9, mp4a.E1"
            bandwidth="128000"
            id="128">
            <SegmentInfo duration="PT10S">
                <BaseURL>"rep2"</BaseURL>
                <InitialisationSegmentURL sourceURL="seg-init.3gp"/>
                <Url sourceURL="seg-1.3gp"/>
                <Url sourceURL="seg-2.3gp"/>
                <Url sourceURL="seg-3.3gp"/>
            </SegmentInfo>
        </Representation>
    </Period>
    <Period start="PT30S">
        <SegmentInfoDefault
            duration="PT10S"
            sourceUrlTemplate="http://example.com/$RepresentationId$/$Index$.3gp"/>
        <Representation
            mimeType="video/3gpp; codecs=mp4v.20.9, mp4a.E1"
            bandwidth="256000"
            id="1">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="seg-init-1.3gp"/>
            </SegmentInfo>
        </Representation>
        <Representation
            mimeType="video/3gpp; codecs=mp4v.20.9, mp4a.E1"
            bandwidth="128000"
            id="2">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="seg-init-2.3gp"/>
            </SegmentInfo>
        </Representation>
```

```
    </Period>
</MPD>
```

D.2 Live Service

Table D.2 provides an example MPD for a live service.

Table D.2: Example MPD for a Live Service

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
    type="Live"
    minBufferTime="PT3S"
    availabilityStartTime="2010-04-26T08:45:00-08:00"
    minimumUpdatePeriodMPD="PT5M0S"
    timeShiftBufferDepth="PT1H30M0S"
    xsi:schemaLocation="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009 3GPP-Rel10-MPD.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
    <ProgramInformation moreInformationURL="http://www.example.com">
        <Title>Example 3: 3GPP SA4 Meeting in Vancouver as Live Broadcast</Title>
        <Source>3GPP</Source>
    </ProgramInformation>
    <Period start="PT0S">
        <Representation id="Ad-QVGA" mimeType='video/3gpp' codecs="avc1.42E00B"
bandwidth="10000" width="320" height="240">
            <SegmentInfo duration="PT60S">
                <InitialisationSegmentURL sourceURL="http://www.ad-server.com/1-day-
black/QVGA/0.3gp"/>
                <UrlTemplate sourceURL="http://www.ad-server.com/1-day-
black/QVGA/$Index$.3gp"/>
            </SegmentInfo>
        </Representation>
    </Period>
    <Period start="PT15M0S">
        <SegmentInfoDefault duration="PT10S"
sourceURLTemplate="http://www.example.com/Period-2010-04-26T08-45-00/rep-
$RepresentationID$/seg-$Index$.3gp"/>
        <Representation id="QVGA-LQ" mimeType='video/3gpp' codecs="avc1.42E00C, mp4a.40.2"
bandwidth="192000" width="320" height="240">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="http://www.example.com/rep-QVGA-LQ/seg-
init.3gp"/>
            </SegmentInfo>
        </Representation>
        <Representation id="QVGA-HQ" mimeType='video/3gpp' codecs="avc1.42E00C, mp4a.40.2"
bandwidth="384000" width="320" height="240">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="http://www.example.com/rep-QVGA-HQ/seg-
init.3gp"/>
            </SegmentInfo>
        </Representation>
        <Representation id="VGA-LQ" mimeType='video/3gpp' codecs="avc1.64001E, mp4a.40.2"
bandwidth="512000" width="640" height="480">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="http://www.example.com/rep-VGA-LQ/seg-
init.3gp"/>
            </SegmentInfo>
        </Representation>
        <Representation id="VGA-HQ" mimeType='video/3gpp' codecs="avc1.64001E, mp4a.40.2"
bandwidth="1024000" width="640" height="480">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="http://www.example.com/rep-VGA-HQ/seg-
init.3gp"/>
            </SegmentInfo>
        </Representation>
    </Period>
    <Period start="PT2H01M22.12S">
        <SegmentInfoDefault duration="PT10S" sourceURLTemplate="http://www.ad-
server.com/15min-Ads/$RepresentationID$/$Index$.3gp"/>
        <Representation id="QVGA" mimeType='video/3gpp' codecs="avc1.42E00C, mp4a.40.2"
bandwidth="256000" width="320" height="240">
            <SegmentInfo>
```

```
                <InitialisationSegmentURL sourceURL="http://www.ad-server.com/15min-
Ads/QVGA/0.3gp"/>
            </SegmentInfo>
        </Representation>
        <Representation id="VGA" mimeType='video/3gpp' codecs="avc1.64001E, mp4a.40.2"
bandwidth="512000" width="640" height="480">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="http://www.ad-server.com/15min-
Ads/VGA/0.3gp"/>
            </SegmentInfo>
        </Representation>
    </Period>
    <Period start="PT2H16M22.12S">
        <SegmentInfoDefault duration="PT10S"
sourceURLTemplate="http://www.example.com/Period-2010-04-26T11-01-22/rep-
$RepresentationID$/seg-$Index$.3gp"/>
        <Representation id="QVGA-LQ" mimeType='video/3gpp' codecs="avc1.42E00C, mp4a.40.2"
bandwidth="192000" width="320" height="240"/>
        <Representation id="QVGA-HQ" mimeType='video/3gpp' codecs="avc1.42E00C, mp4a.40.2"
bandwidth="384000" width="320" height="240"/>
        <Representation id="VGA-LQ" mimeType='video/3gpp' codecs="avc1.64001E, mp4a.40.2"
bandwidth="512000" width="640" height="480"/>
        <Representation id="VGA-HQ" mimeType='video/3gpp' codecs="avc1.64001E, mp4a.40.2"
bandwidth="1024000" width="640" height="480"/>
    </Period>
</MPD>
```

D.3 MPD Assembly

Table D.3 provides an example MPD with reference to external Period element as provided in Table D.4. An equivalent MPD to the one in Table D.3 after dereferencing with the **`Period`** element in Table D.4 is shown in Table D.5.

Table D.3: Example MPD with reference to external Period element

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
    type="Live"
    minBufferTime="PT3S"
    availabilityStartTime="2010-04-26T08:45:00-08:00"
    minimumUpdatePeriodMPD="PT5M0S"
    timeShiftBufferDepth="PT1H30M0S"
    xsi:schemaLocation="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009 3GPP-Rel10-MPD.xsd
http://www.w3.org/1999/xlink xlink.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
    <ProgramInformation moreInformationURL="http://www.example.com">
        <Title>Example 3: 3GPP SA4 Meeting in Vancouver as Live Broadcast</Title>
        <Source>3GPP</Source>
    </ProgramInformation>
    <Period start="PT0S" duration="PT2H01M22.12S">
        <Representation id="Ad-QVGA" mimeType="video/3gpp" codecs="avc1.42E00B"
bandwidth="10000" width="320" height="240">
            <SegmentInfo duration="PT60S">
                <InitialisationSegmentURL sourceURL="http://www.ad-server.com/1-day-
black/QVGA/0.3gp"/>
                <UrlTemplate sourceURL="http://www.ad-server.com/1-day-
black/QVGA/$Index$.3gp"/>
            </SegmentInfo>
        </Representation>
    </Period>
    <Period xlink:ref="http://www.example.com/Period.xml"/>
    <Period start="PT2H16M22.12S">
        <SegmentInfoDefault duration="PT10S"
sourceURLTemplate="http://www.example.com/Period-2010-04-26T11-01-22/rep-
$RepresentationID$/seg-$Index$.3gp"/>
        <Representation id="QVGA-LQ" mimeType="video/3gpp" codecs="avc1.42E00C, mp4a.40.2"
bandwidth="192000" width="320" height="240"/>
        <Representation id="QVGA-HQ" mimeType="video/3gpp" codecs="avc1.42E00C, mp4a.40.2"
bandwidth="384000" width="320" height="240"/>
        <Representation id="VGA-LQ" mimeType="video/3gpp" codecs="avc1.64001E, mp4a.40.2"
bandwidth="512000" width="640" height="480"/>
        <Representation id="VGA-HQ" mimeType="video/3gpp" codecs="avc1.64001E, mp4a.40.2"
bandwidth="1024000" width="640" height="480"/>
```

```
    </Period>
</MPD>
```

Table D.4: External Period

```
<?xml version="1.0" encoding="UTF-8"?>
<Period>
    <SegmentInfoDefault duration="PT10S" sourceURLTemplate="http://www.example.com/Period-
06/rep-$RepresentationID$/seg-$Index$.3gp"/>
    <Representation id="QVGA-LQ" mimeType="video/3gpp" codecs="avc1.42E00C, mp4a.40.2"
bandwidth="192000" width="320" height="240">
        <SegmentInfo>
            <InitialisationSegmentURL sourceURL="http://www.example.com/rep-QVGA-LQ/seg-
init.3gp"/>
        </SegmentInfo>
    </Representation>
    <Representation id="QVGA-HQ" mimeType="video/3gpp" codecs="avc1.42E00C, mp4a.40.2"
bandwidth="384000" width="320" height="240">
        <SegmentInfo>
            <InitialisationSegmentURL sourceURL="http://www.example.com/rep-QVGA-HQ/seg-
init.3gp"/>
        </SegmentInfo>
    </Representation>
</Period>
```

Table D.5: Equivalent MPD to the one in Table D.3 after dereferencing with the element in Table D.4

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
    type="Live"
    minBufferTime="PT3S"
    availabilityStartTime="2010-04-26T08:45:00-08:00"
    minimumUpdatePeriodMPD="PT5M0S"
    timeShiftBufferDepth="PT1H30M0S"
    xsi:schemaLocation="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009 3GPP-Rel10-MPD.xsd
http://www.w3.org/1999/xlink xlink.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
    <ProgramInformation moreInformationURL="http://www.example.com">
        <Title>Example 3: 3GPP SA4 Meeting in Vancouver as Live Broadcast</Title>
        <Source>3GPP</Source>
    </ProgramInformation>
    <Period start="PT0S" duration="PT2H01M22.12S">
        <Representation id="Ad-QVGA" mimeType="video/3gpp" codecs="avc1.42E00B"
bandwidth="10000" width="320" height="240">
            <SegmentInfo duration="PT60S">
                <InitialisationSegmentURL sourceURL="http://www.ad-server.com/1-day-
black/QVGA/0.3gp"/>
                <UrlTemplate sourceURL="http://www.ad-server.com/1-day-
black/QVGA/$Index$.3gp"/>
            </SegmentInfo>
        </Representation>
    </Period>
    <Period/>
    <Period start="PT2H01M22.12S">
        <SegmentInfoDefault duration="PT10S"
sourceURLTemplate="http://www.example.com/Period-06/rep-$RepresentationID$/seg-$Index$.3gp"/>
        <Representation id="QVGA-LQ" mimeType="video/3gpp" codecs="avc1.42E00C, mp4a.40.2"
bandwidth="192000" width="320" height="240">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="http://www.example.com/rep-QVGA-LQ/seg-
init.3gp"/>
            </SegmentInfo>
        </Representation>
        <Representation id="QVGA-HQ" mimeType="video/3gpp" codecs="avc1.42E00C, mp4a.40.2"
bandwidth="384000" width="320" height="240">
            <SegmentInfo>
                <InitialisationSegmentURL sourceURL="http://www.example.com/rep-QVGA-HQ/seg-
init.3gp"/>
            </SegmentInfo>
        </Representation>
    </Period>
    <Period start="PT2H16M22.12S">
        <SegmentInfoDefault duration="PT10S"
sourceURLTemplate="http://www.example.com/Period-2010-04-26T11-01-22/rep-
```

```
$RepresentationID$/seg-$Index$.3gp"/>
        <Representation id="QVGA-LQ" mimeType="video/3gpp" codecs="avc1.42E00C, mp4a.40.2"
bandwidth="192000" width="320" height="240"/>
        <Representation id="QVGA-HQ" mimeType="video/3gpp" codecs="avc1.42E00C, mp4a.40.2"
bandwidth="384000" width="320" height="240"/>
        <Representation id="VGA-LQ" mimeType="video/3gpp" codecs="avc1.64001E, mp4a.40.2"
bandwidth="512000" width="640" height="480"/>
        <Representation id="VGA-HQ" mimeType="video/3gpp" codecs="avc1.64001E, mp4a.40.2"
bandwidth="1024000" width="640" height="480"/>
    </Period>
</MPD>
```

D.4 MPD Deltas

In the following MPD example, the content is 30 minutes in duration. There are 3 Periods, each of 10 minutes duration. Each Period has 3 Representations and each Representation is contained within one 3gp file. Each Representation has audio encoded with Low Complexity-AAC. One Representation of each Period (p1rep1.3gp, p2rep1.3gp, and p3rep1.3gp) has video resolution 320x240 encoded with H.264 baseline profile level 1.1. Another Representation of each Period (p1rep2.3gp, p2rep2.3gp, and p3rep2.3gp) has resolution 320x240 encoded with H.264 baseline profile level 1.3. Finally, a third representation in each period (p1rep3.3gp, p2rep3.3gp, and p3rep3.3gp) has resolution 480x240 encoded with H.264 baseline profile level 2.1. One Representation of each Period has bandwidth of 239 kbps, a second representation has bandwidth of 478 kbps, and a third representation has bandwidth of 892 kbps.

Since each represention is contained in one file, the initialization segments and the media segments for a representation are accessed with byte ranges. Each 'Url' element in the MPD contains a 'range' and the corresponding byte range for the initialization segment or media segment. For the example each segment of all representations is 10 seconds in duration.

Line numbers of the MPD in the example are shown for clarity, although these would not be present in the MPD.

EXAMPLE 1 (add)

The change of adding the "Url" element for the last Segment to the Representation of the third Period with 239K bandwidth can be described as

```
517a
                <Url range="17339554-17642841"/>
.
```

The line number of the MPD where the delta is applied is 517. The following line is added:

EXAMPLE 2 (replace)

Consider the change of replacing the line containing the **`DeltaSupport`** element in the next MPD.

```
20c
      <DeltaSupport sourceURL="delta2.mpdd" availabilityDuration="120s"/>
.
```

EXAMPLE 3(delete)

If lines 8 through 10 of the original MPD are deleted and not present in the updated MPD, the delta to express this is:

```
8,10d
.
```

Below is what the MPD looks like after 30 minutes. In this case, the MPD is updated approximately every 10 seconds.

```
1<?xml version="1.0" encoding="UTF-8"?>
2<MPD
3       type="Live"
4       availabilityStartTime="2010-07-01T05:00:00Z"
5       availabilityEndTime="2010-07-08T05:00:00Z"
6       mediaPresentationDuration="PT2H"
7       minimumUpdatePeriodMPD="PT10S"
8       minBufferTime="PT10S"
```

```
9       timeShiftBufferDepth="PT30M"
10      baseUrl="http://www.example.com/"
11      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
12      xsi:schemaLocation="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2010 3GPP-MPD-r2.xsd"
13      xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2010">
14
15 <ProgramInformation moreInformationURL="http://www.example.com">
16  <Title>Example</Title>
17  <Source>Example</Source>
18  <Copyright>Example</Copyright>
19 </ProgramInformation>
20 <DeltaSupport sourceURL="delta1.mpdd" availabilityDuration="120s"/>
21 <Period start="PT0S" segmentAlignmentFlag="true" bitstreamSwitchingFlag="true">
22    <Representation  Id="0" bandwidth="239000" width="320" height="240" lang="en"
mimeType="video/3gpp; codecs=avc1.42E00b, mp4a.40.2" >
23       <SegmentInfo duration="PT10S" baseURL="p1rep1.3gp">
24          <InitialisationSegmentURL range="0-985" />
25                               <Url  range="986-293761" />
26                               <Url  range="293762-592501" />
                                                  .
                                                  .
                                                  .

84                               <Url  range="17600065-17894640" />
85       </SegmentInfo>
86    </Representation>
87    <Representation Id="1" bandwidth="478000" width="320" height="240" lang="en"
mimeType="video/3gpp; codecs=avc1.42E00d, mp4a.40.2" >
88       <SegmentInfo duration="PT10S" baseURL="p1rep2.3gp">
89          <InitialisationSegmentURL range="0-985" />
90                               <Url  range="986-586538" />
91                               <Url  range="586539-1184019" />
                                                  .
                                                  .
                                                  .
149                              <Url  range="35199171-35788323" />
150      </SegmentInfo>
151    </Representation>
152    <Representation Id="2" bandwidth="892000" width="480" height="240" lang="en"
mimeType="video/3gpp; codecs=avc1.42E015, mp4a.40.2" >
153      <SegmentInfo duration="PT10S" baseURL="p1rep3.3gp">
154           <InitialisationSegmentURL range="0-985" />
155                              <Url  range="986-1093691" />
156                              <Url  range="1093692-2208656" />
                                                  .
                                                  .
                                                  .
214                              <Url  range="65684646-66784068" />
215      </SegmentInfo>
216   </Representation>
217</Period>
217<Period start="PT10M0S" segmentAlignmentFlag="true" bitstreamSwitchingFlag="true">
218 <Representation  Id="0" bandwidth="239000" width="320" height="240" lang="en"
mimeType="video/3gpp; codecs=avc1.42E00b, mp4a.40.2" >
219      <SegmentInfo duration="PT10S" baseURL="p2rep1.3gp">
220         <InitialisationSegmentURL range="0-985" />
221                              <Url  range="986-296011" />
222                              <Url  range="296012-595787" />
                                                  .
                                                  .
                                                  .

280                               <Url  range="17647666-17946154" />
281      </SegmentInfo>
282 </Representation>
283 <Representation Id="1" bandwidth="478000" width="320" height="240" lang="en"
mimeType="video/3gpp; codecs=avc1.42E00d, mp4a.40.2" >
284      <SegmentInfo duration="PT10S" baseURL="p2rep2.3gp">
285         <InitialisationSegmentURL range="0-985" />
286                              <Url  range="986-591037" />
                                 <Url  range="591038-1190590" />
                                                  .
                                                  .
                                                  .
385                              <Url  range="35294377-35891354" />
386      </SegmentInfo>
387  </Representation>
388  <Representation Id="2" bandwidth="892000" width="480" height="240" lang="en"
```

```
mimeType="video/3gpp; codecs=avc1.42E015, mp4a.40.2" >
389      <SegmentInfo duration="PT10S" baseURL="p2rep3.3gp">
390         <InitialisationSegmentURL range="0-985" />
391                              <Url  range="986-1102088" />
392                              <Url  range="1102089-2220920" />
                                                  .
                                                  .
                                                  .
450                              <Url  range="65862331-66976355" />
451      </SegmentInfo>
452   </Representation>
453</Period>
454<Period start="PT20M0S" segmentAlignmentFlag="true" bitstreamSwitchingFlag="true">
455   <Representation  Id="0" bandwidth="239000" width="320" height="240" lang="en"
mimeType="video/3gpp; codecs=avc1.42E00b, mp4a.40.2" >
456      <SegmentInfo duration="PT10S" baseURL="p3rep1.3gp">
457         <InitialisationSegmentURL range="0-985" />
458                              <Url  range="986-302469" />
459                              <Url  range="302470-597839" />
                                                  .
                                                  .
                                                  .
517                              <Url  range="17040002-17339553" />
518      </SegmentInfo>
519   </Representation>
520   <Representation Id="1" bandwidth="478000" width="320" height="240" lang="en"
mimeType="video/3gpp; codecs=avc1.42E00d, mp4a.40.2" >
521      <SegmentInfo duration="PT10S" baseURL="p3rep2.3gp">
522         <InitialisationSegmentURL range="0-985" />
523                              <Url  range="986-603953" />
524                              <Url  range="603954-1194693" />
                                                  .
                                                  .
                                                  .
582                              <Url  range="34079046-34678149" />
583      </SegmentInfo>
584   </Representation>
585   <Representation Id="2" bandwidth="892000" width="480" height="240" lang="en"
mimeType="video/3gpp; codecs=avc1.42E015, mp4a.40.2" >
586      <SegmentInfo duration="PT10S" baseURL="p3rep3.3gp">
587         <InitialisationSegmentURL range="0-985" />
588                              <Url  range="986-1126190" />
589                              <Url  range="1126191-2228575" />
                                                  .
                                                  .
                                                  .
647                               <Url  range="63594383-64712374" />
648      </SegmentInfo>
649   </Representation>
650 </Period>
651</MPD>
```

Since the value of `@sourceURL` in the above MPD is 'delta1.mpdd ', delta1.mpdd is an empty file at the time of publication of the above MPD.

The following file is delta1.mpdd after the next MPD update. Notice that clients have access to the new value of `@sourceURL` referenced by the latest MPD via the delta.

```
647a
               <Url range="64712375-65844316"/>
.
582a
               <Url range="34678150-35284727"/>
.
517a
               <Url range="17339554-17642841"/>
.
20c
      <DeltaSupport sourceURL="delta2.mpdd" availabilityDuration="120s"/>
.
```

At the next MPD update, 'delta1.mpdd' would contain the cumulative update for 2 MPD updates.

```
647a
                <Url range="64712375-65844316"/>
                <Url range="65844317-66966044"/>
.
582a
                <Url range="34678150-35284727"/>
                <Url range="35284728-35885833"/>
.
517a
                <Url range="17339554-17642841"/>
                <Url range="17642842-17943394"/>
.
20c
      <DeltaSupport sourceURL="delta3.mpdd" availabilityDuration="120s"/>
.
```

Annex E (informative): Mapping MPD structure and semantics to SMIL

E.1 General

The mapping presented in this Annex allows transformation of the MPD table and the XML schema defined in Annex B to a SMIL-based syntax. This transformation may be effected automatically, for instance, using XSLT, at the client or the server. The MPD structure and semantics will be retained in the SMIL-based syntax.

The first 3 columns of Table E.1 below contain the elements and attributes present in the MPD. Column 1 contains an MPD element, column 2 lists its children elements and column 3 lists its attributes.

Column 4 indicates how elements/attributes from this structure can be mapped to elements/attributes in 3GPP SMIL. That is, it indicates which 3GPP SMIL attributes/elements can be used to provide equivalent functionality.

Note that '3GPP SMIL' as used in the present document refers to the 3GPP SMIL Language profile defined in TS 26.246 [6].

In some cases to match the semantics of the attributes and elements, new attributes/elements that are not defined in 3GPP SMIL, are required. Column 5 lists these attributes or elements. These would be added to 3GPP SMIL as extensions indicated by the '3g9' identifier defined in the same namespace as that for 3GP-DASH in clause 8.2.2, viz., xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009".

In many cases deployments that do not use the new elements and attributes from column 5 are feasible– these elements and attributes are either optional or existing SMIL constructs can be used as an alternative (e.g. playlists can be used instead of templates). System deployers should only use constructs in column 4 if they want compatibility with legacy 3GPP SMIL clients. If support for extension elements and attributes has been added to 3GPP SMIL clients, deployments can leverage constructs in column 4 as well as 5.

Table E.1: Mapping MPD structure, semantics and syntax to SMIL

Element	Child Elements	Attribute	Mapping to 3GPP SMIL	Extension to 3GPP SMIL
MPD	**BaseURL, Period, ProgramInformation**		body	
		@type		@type
		@availabilityStartTime		@availabilityStartTime
		@availabilityEndTime		@availabilityEndTime
		@mediaPresentationDuration		@mediaPresentationDuration
		@minimumUpdatePeriodMPD		@minimumUpdatePeriodMPD
		@minBufferTime		@minBufferTime
		@timeShiftBufferDepth		@timeShiftBufferDepth
Period	**Representation, SegmentInfoDefault**		seq, switch	
		@start	begin	
		@id	id	
		@segmentAlignmentFlag		@segmentAlignmentFlag
		@bitstreamSwitchingFlag		@bitStreamSwitchingFlag
ProgramInformation	**Source, Copyright, Title**		meta	
		@moreInformationURL		@moreInformationURL
BaseURL				**BaseURL**
Location				**Location**
Representation	**SegmentInfo, ContentProtection, TrickMode**		seq	
		@id	id	
		@bandwidth	systemBitrate	
		@width, @height	systemScreenSize	
		@lang	systemLanguage	
		@mimeType	systemComponent	
		@group		@group
		@startWithRAP		@startWithRAP
		@qualityRanking		@qualityRanking
SegmentInfo	**InitialisationSegmentUrl, Url, UrlTemplate BaseURL**		par, seq When track alignment across Segments cannot be guaranteed, par should be used with each children URL containing begin and dur attributes.	**SegmentInfo** element with playback semantics identical to those defined in clause 12.2. That is, all children elements of **SegmentInfo** are time-continuous across boundaries of consecutive Media Segments within one Representation.
		@duration	dur specified for each Url in playlist, possibly in conjunction with begin	
		@startIndex		@startIndex
InitialisationSegmentURL			See **Url** below. To identify initialization segments, either dur can be set to '0' or type can be set to 'init'	

Element	Child Elements	Attribute	Mapping to 3GPP SMIL	Extension to 3GPP SMIL
Url			MEDIA-ELMS (ref, video, audio, etc.) as defined in 3GPP SMIL along with all attributes defined for those elements	
		@sourceURL	src (only allows absolute URIs).	@sourceURL with URI resolution semantics defined in clause 8.2.3. This attribute is also defined for the **MPD**, **SegmentInfo**, **SegmentInfoDefault** and **Url** elements
		@range		range
UrlTemplate			**Url** playlists may be used as an alternative to urlTemplate	**UrlTemplate** (along with attributes defined for UrlTemplate)
SegmentInfoDefault	**UrlTemplate**		This element may be skipped and information provided directly at SegmentInfo level instead	**SegmentInfoDefault**
		@duration		See @duration in Segment
		@startIndex		@startIndex
		@sourceUrlTemplate		@sourceUrlTemplate
ContentProtection	**SchemeInfo**		For 3GP files, content protection may be achieved through mechanisms defined in 3GP file format	**ContentProtection**
		@schemeId		@schemeId

The examples below illustrate the use of a SMIL-based syntax. The examples include 3GPP SMIL constructs as well as extensions to 3GPP SMIL.

E.2 Examples

E.2.1 Example 1: MPD for on-demand content with multiple Periods and alternate Representations

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
      xmlns:3g9="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
<body>

<!-- Period 1-->
<seq begin='0s'>
<!-- alternate set of Representations available during this Period -->
    <switch>
        <!-- low bitrate  Representation with 15-sec Segments -->
        <seq systemBitrate='128000'>
            <video dur='0s'  src="http://server/path/rep1/clip_init.3gp"/>
            <par>
                    <video begin='0s' dur='15s' src="http://server/path/rep1/clip_1.3gp"/>
                    <video begin='15s' dur='15s' src="http://server/path/rep1/clip_1.3gp"/>
                    ...
                    <video begin='585s' dur='15s' src="http://server/path/rep1/clip_40.3gp"/>
            </par>
       </seq>
        <!-- mid bitrate Representation with 30-sec Segments -->
        <seq systemBitrate='256000'>
            <video dur='0s' src="http://server/path/rep2/clip_init.3gp'/>
            <par>
                    <video begin='0s' dur='30s' src="http://server/path/rep2/clip_1.3gp"/>
                    ...
                    <video begin='570s' dur='30s' src="http://server/path/rep2/clip_20.3gp"/>
```

```
            </par>
        </seq>
         <!-- high bitrate Representation with 30-sec Segments-->
         <seq systemBitrate='512000'>
            <video type='init' dur='0s' src="http://server/path/rep3/clip_init.3gp'/>
            <par>
                    <video begin='0s' dur='30s' src="http://server/path/rep3/clip_1.3gp"/>
                    ...
                    <video begin='570s' dur='30s' src="http://server/path/rep3/clip_20.3gp"/>
            </par>
         </seq>
    </switch>
</seq> <!-- end of Period 1 -->

<!-- Period 2 begins 10 minutes after presentation start -->
<seq begin='600s'>
    <switch>
        <!-- english ad -->
        <seq systemLanguage='en' >
            <seq>
                    <video src='http://adserver/getad.php?id=1'/>
            </seq>
        </seq>
        <!-- french ad -->
        <seq systemLanguage='fr' >
            <seq>
                    <video src='http://adserver/getad.php?id=2'/>
            </seq>
        </seq>
    </switch>
</seq>

<!--start  of Period 3. Note that mid bitrate Representation is missing during this Period. Also the
server used to deliver Segments is   different than the one in the previous Period. The use of URL
resolution as defined by sourceURL is illustrated -->
<seq begin='630s'  3g9:sourceURL='http://new-server/new-path/'>
    <switch>
        <!-- low bitrate  -->
        <seq systemBitrate='128000' 3g9:baseURL='rep1/'>
            <!-- no initialisation Segment -->
            <par>
                    <video begin='0s' dur='15s' 3g9:sourceURL="clip41.3gp'/>
                    ...
            </par>
        </seq>
        <!-- high bitrate -->
        <seq systemBitrate='512000' 3g9:baseURL='rep3/'>
            <video type='init' dur='0s' 3g9:sourceURL='clip2x_init.3gp'/>
            <par>
                    <!-- URLs can include a byte range -->
                    <video begin='0s' dur='30s' 3g9:sourceURL='clip2x.3gp' 3g9:range='500-2000'/>
                    <video begin='30s' dur='30s' 3g9:sourceURL='clip2x.3gp' 3g9:range='2001-2500'/>
                    …
            </par>
        </seq>
    </switch>
 </seq>


 <!-- more Periods can go here as required -->
  …
</body>
</smil>
```

E.2.2 Example 2: MPD for live content

MPD that includes *availabilityStartTime* and *availabilityEndTime* extensions to enforce lifetime and the minimumUpdatePeriodMPD extension to help clients choose an update period. Note that Segment format used in the example is MPEG-2 TS.

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
      xmlns:3g9="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
<body 3g9: minimumUpdatePeriodMPD ='120s' 3g9:availabilityStartTime='2010-01-27T13:00Z'
3g9:availabilityEndTime='2010-01-27T15:00Z'3g9:minBufferTime='10s' 3g9:type='live'>
<!-- start of a Period -->
<seq begin='0s'>
```

```
    <!-- a single Representation -->
    <seq>
        <par>
                <video begin='0s' dur='10s' src='http://server/live_clip_1.m2ts'/>
                …
                <video begin='110s' dur='10s' src='http://server/live_clip_12.m2ts'/>
        </par>
    </seq> <!-- end of Representation -->
  </seq> <!-- end of Period -->
</body>
</smil>
```

Annex F (normative): OMA DM QoE Management Object

As an alternative to configuring the QoE reporting for each session via MPD, OMA-DM can be used to specify the QoE configuration. If such an OMA-DM QoE configuration has been specified, it shall be evaluated by the client for all subsequent sessions.

For the OMA-DM QoE configuration the parameters are specified according to the following Managed Object (MO), and represents the same information as specified in section 10.4 and 10.5. Version numbering is included for possible extension of the MO.

The Management Object Identifier shall be: `urn:oma:mo:ext-3gpp-pss-dash-qoe:1.0`.

Protocol compatibility: The MO is compatible with OMA Device Management protocol specifications, version 1.2 and upwards, and is defined using the OMA DM Device Description Framework as described in the Enabler Release Definition OMA-ERELD _DM-V1_2 [22].

The nodes and leaf objects as provided in Figure F.1 shall be contained under the 3GPP_PSS_DASH_QOE node if a client supports the feature described in this clause.

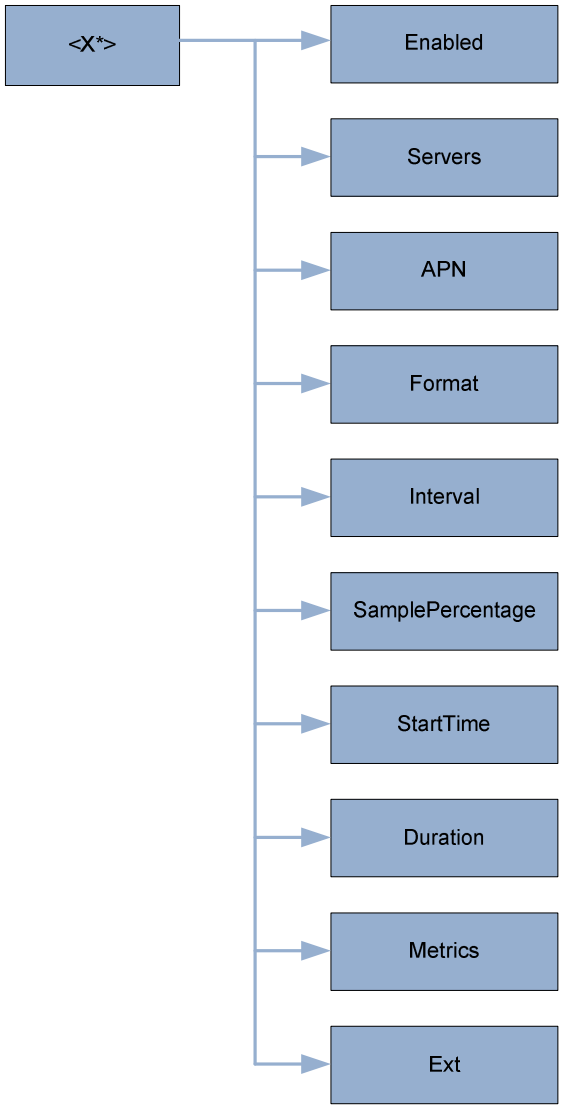

Figure F.1: Nodes and leaf objects

Node: /<X>

This interior node specifies the unique object id of a QoE metrics management object. The purpose of this interior node is to group together the parameters of a single object.

- Occurrence: ZeroOrOne
- Format: node
- Minimum Access Types: Get

The following interior nodes shall be contained if the client supports the QoE Management Object.

/<X>/Enabled

This leaf indicates if QoE reporting is requested by the provider.

- Occurrence: One
- Format: bool
- Minimum Access Types: Get

/<X>/Servers

This leaf contains a space-separated list of servers to which the QoE reports are transmitted. It is URI addresses, e.g. http://qoeserver.operator.com. In case of multiple servers, the client randomly selects one of the servers from the list, with uniform distribution.

- Occurrence: One
- Format: chr
- Minimum Access Types: Get
- Values: URI of the servers to receive the QoE report.

/<X>/APN

This leaf contains the Access Point Name that should be used for establishing the PDP context on which the QoE metric reports will be transmitted. This may be used to ensure that no costs are charged for QoE metrics reporting. If this leaf is not defined then any QoE reporting is done over the default access point.

- Occurrence: ZeroOrOne
- Format: chr
- Minimum Access Types: Get
- Values: The Access Point Name

/<X>/Format

This leaf specifies the format of the report.If this leaf is not defined the QoE reports shall be sent uncompressed.

- Occurrence: ZeroOrOne
- Format: chr
- Minimum Access Types: Get
- Values: 'uncompressed', 'gzip'

/<X>/Interval

This leaf specifies how often QoE reports shall be sent. If this leaf is not defined only one QoE report shall be sent after the complete session.

- Occurrence: ZeroOrOne
- Format: int
- Minimum Access Types: Get
- Values: seconds

/<X>/SamplePercentage

This leaf specifies the percentage of sessions for which QoE metrics shall be reported. The client evaluates a random number at start of each session to determine if reporting shall be done for the specific session. If this leaf is not defined QoE reports are sent for every session.

- Occurrence: ZeroOrOne
- Format: float
- Minimum Access Types: Get
- Values: 0.0-100.0.

/<X>/StartTime

This leaf specifies when collection of QoE metrics shall start. It is specified in seconds and is relative to the start of the session. If this leaf is not defined, the QoE collection shall be done from the start of the session.

- Occurrence: ZeroOrOne
- Format: int
- Minimum Access Types: Get
- Values: seconds

/<X>/Duration

This leaf specifies for how long QoE collection shall be done. It is specified in seconds and is relative to the start time of QoE collection. If this leaf is not defined QoE collection shall be done until the end of the session.

- Occurrence: ZeroOrOne
- Format: int
- Minimum Access Types: Get
- Values: seconds.

/<X>/Metrics

This leaf specifies a list of white-space separated metrics which shall be reported, and follows the same syntax as specified for the "@metrics" attribute in Table 32. If this leaf is not defined no QoE reporting shall be done.

- Occurrence: ZeroOrOne
- Format: chr
- Minimum Access Types: Get
- Values: Metrics as specified in section 10.4.

/<X>/Ext

The Ext node is an interior node where the vendor specific information can be placed (vendor includes application vendor, device vendor etc.). Usually the vendor extension is identified by vendor specific name under the ext node. The tree structure under the vendor identified is not defined and can therefore include one or more un-standardized sub-trees.

- Occurrence: ZeroOrOne
- Format: node
- Minimum Access Types: Get

Annex G (normative): File format extensions for 3GPP DASH support

G.1 Introduction

This clause documents extensions to the ISO base media file format [11] for the support of 3GPP DASH. It is expected that these boxes will be integrated in an updated version of ISO/IEC 14496-12 [11].

G.2 Level Assignment Box

G.2.1 Definition

Box Type: `leva"
Container: Movie Extends Box (`mvex")
Mandatory: No
Quantity: Zero or one

Levels specify subsets of the file. Samples mapped to level n may depend on any samples of levels m, where m <= n, and shall not depend on any samples of levels p, where p > n.

Levels cannot be specified for the initial movie. When the Level Assignment box is present, it applies to all movie fragments subsequent to the initial movie.

For the context of the Level Assignment box, a fraction is defined to consist of one or more Movie Fragment boxes and the associated Media Data boxes, possibly including only an initial part of the last Media Data Box. Within a fraction, data for each level shall appear contiguously. Data for levels within a fraction shall appear in increasing order of level value. All data in a fraction shall be assigned to levels.

NOTE: In the context of 3G DASH, each subsegment indexed within a Subsegment Index box is a fraction.

The Level Assignment box provides a mapping from features, such as temporal sub-sequences, to levels. A feature can be specified through a track or a sample grouping of a track.

The following assignment_types are defined; `assignment_type` values greater than 4 are reserved, while the semantics for the other values are specified as follows.

- 0: sample groups are used to specify levels, i.e. i.e. samples mapped to different sample group description indexes of a particular sample grouping lie in different levels within the identified track; other tracks are not affected and must have all their data in precisely one level;
- 1: as for assignment_type 0 except assignment is by a parameterized sample group;
- 2, 3: level assignment is by track (see the Subsegment Index Box for the difference in processing of these levels)

The sequence of assignment_types is restricted to be a set of zero or more of type 2 or 3, followed by zero or more of exactly one type.

G.2.2 Syntax

```
aligned(8) class LevelAssignmentBox extends FullBox("leva", 0, 0) { unsigned int(8) level_count;
    for (j=1; j <= level_count; j++) {
        unsigned int(32)    track_id;
        unsigned int(1) padding_flag;
        unsigned int(7) assignment_type;
        if (assignment_type == 0)
            unsigned int(32)    grouping_type;
        else if (assignment_type == 1) {
```

```
            unsigned int(32)    grouping_type;
            unsigned int(32)    grouping_type_parameter;
        }
        else if (assignment_type == 2) {} // no further syntax elements needed
        else if (assignment_type == 3) {} // no further syntax elements needed  }
}
```

G.2.3 Semantics

`level_count` specifies the number of levels each fraction is grouped into. `level_count` shall be greater than or equal to 2.

`track_id` for loop entry j specifies the track identifier of the track assigned to level j.

`padding_flag` equal to 1 indicates that a conforming fraction can be formed by concatenating any positive integer number of levels within a fraction and padding the last Media Data box by zero bytes up to the full size that is indicated in the header of the last Media Data box. The semantics of `padding_flag` equal to 0 are unspecified.

`assignment_type` indicates the mechanism used to specify the assignment to a level. `assignment_type` values greater than 3 are reserved, while the semantics for the other values are specified as follows.

`grouping_type` and `grouping_type_parameter`, if present, specify the sample grouping used to map sample group description entries in the Sample Group Description box to levels. Level n contains the samples that are mapped to the sample group description entry having index n in the Sample Group Description box having the same values of `grouping_type` and `grouping_type_parameter`, if present, as those provided in this box.

G.3 Subsegment Index Box

G.3.1 Definition

Box Type: `ssix"
Container: File
Mandatory: No
Quantity: Zero or more

The Subsegment Index box ('ssix') provides a mapping from levels (as specified by the Level Assignment box) to byte ranges of the indexed subsegment. In other words, this box provides a compact index for how the data in is ordered according to levels into partial sub-segments. It enables a client to easily access data for partial subsegments by downloading ranges of data in the subsegment.

Each byte in the subsegment shall be assigned to a level. If the range is not associated with any information in the level assignment, then any level that is not included in the level assignment may be used. Each level shall be assigned to exactly one partial sub-segment, i.e. byte ranges for one level shall be contiguous.

Samples of a partial subsegment may depend on any samples of preceding partial subsegments in the same subsegment, but not the other way around. For example, each partial subsegment contains samples having an identical temporal level and partial subsegments appear in increasing temporal level order within the subsegment.

There may be 0 or 1 Subsegment Index boxes per each Segment Index box that does not refer to other Segment Index boxes, i.e. that only indexes subsegments but no segment indexes. A Subsegment Index box, if any, shall be the next box after the associated Segment Index box. A Subsegment Index box documents the subsegment that is indicated in the immediately preceding Segment Index box.

When a partial segment is accessed in this way, for all `assignment_types` other than 3, the final Media Data box may be incomplete, that is, less data is accessed than the length indication of the Media Data Box indicates is present. The length of the Media Data box may need adjusting, or padding used. The padding_flag in the Level Assignment Box indicates whether this missing data can be replaced by zeros. If not, the sample data for samples assigned to levels that are not accessed is not present, and care should be taken not to attempt to process such samples.

NOTE: `assignment_type` equal to 3 may be used, for example, when audio and video movie fragments (including the respective Media Data boxes) are interleaved. The first level can be specified to contain the audio movie fragments (including the respective Media Data boxes), whereas the second level can be specified to contain both audio and video movie fragments (including all Media Data boxes).

G.3.2 Syntax

```
aligned(8) class SubsegmentIndexBox extends FullBox("ssix", 0, 0) {
    unsigned int(32)    subsegment_count;


    for( i=1; i <= subsegment_count; i++)
      unsigned int(8)   ranges_count;
        for ( j=1; j <= ranges_count; j++) {
            unsigned int(8) level;
            unsigned int(24) accumulated_level_size;
      }
}
```

G.3.3 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 `reference_count` (i.e. the number of movie fragment references) in the immediately preceding Segment Index box.

`ranges_count` specifies the number of partial subsegment levels the media data is grouped into. This value shall be greater than or equal to 2.

`range_size`, for any value of j equal to 1, indicates the size of the partial subsegment.

`level` specifies the level to which this partial subsegment is assigned to.

G.4 Temporal level sample grouping

G.4.1 Definition

Many video codecs support temporal scalability where it is possible to extract one or more subsets of frames that can be independently decoded. A simple case is the extraction of I frames for a bitstream with a regular I-frame interval, e.g. IPPPIPPP…, where every 4th picture is an I frame. Also subsets of these I frames can be extracted for even lower frame rates. More elaborate situations with several temporal levels can be constructed using hierarchical B or P frames.

The Temporal Level sample grouping ('tele') provides a codec-independent sample grouping that can be used to group samples (access units) in a track (and potential track fragments) according to temporal level, where samples of one temporal level have no coding dependencies on samples of higher temporal levels. The temporal level equals the sample group description index (taking values 1, 2, 3, etc). The bitstream containing only the access units of from the first temporal level to a higher temporal level remains conforming to the coding standard.

A grouping according to temporal level facilitates easy extraction of temporal subsequences, for instance using the Subsegment Index box in clause G.3.

G.4.2 Syntax

```
class TemporalLevelEntry() extends SampleGroupDescriptionEntry('tele')
{
    bit(1)  level_independently_decodable;
    bit(7)  reserved=0;
}
```

G.4.3 Semantics

The temporal level of samples in a sample group equals to the sample group description index.

`level_independently_decodable` is a flag. 1 indicates that all samples of this level have no coding dependencies on samples of other levels. 0 indicates that no information is provided.

G.5 Producer reference box

G.5.1 Definition

Box Type: `` `prft" ``
Container: File
Mandatory: No
Quantity: Zero or more

The producer reference time box supplies relative wall-clock times at which movie fragments, or files containing movie fragments (such as segments) were produced. When these files are both produced and consumed in real time, this can provide clients with information to enable them to synchronize consumption with the production and thus avoid buffer overflow or underflow.

This box is related to the next movie fragment box that follows it in bitstream order. It must follow any segment type or segment index box (if any) in the segment, and occur before the following movie fragment box (to which it refers). If a segment file contains any producer reference time boxes, then the first of them shall occur before the first movie fragment box in that segment.

The box contains a time value measured on a clock which increments at the same rate as a UTC-synchronized NTP clock, using NTP format. This is associated with a media time for one of the tracks in the movie fragment. That media time should be in the range of times in that track in the associated movie fragment.

G.5.2 Syntax

```
aligned(8) class ProducerReferenceTimeBox extends FullBox("srft", version, 0) {
    unsigned int(32) reference_track_ID;
    unsigned int(64) ntp_timestamp;
    if (version==0)
    {
        unsigned int(32)    media_time;
    } else
    {
        unsigned int(64)    media_time;
    }
}
```

G.5.3 Semantics

`reference_track_ID` provides the `track_ID` for the reference track.
`ntp_timestamp` indicates a UTC time in NTP format corresponding to `decoding_time`.
`media_time` corresponds to the same time as `ntp_timestamp`, but in the time units used for the reference track, and is measured on this media clock as the media is produced. Note that in most cases this timestamp will not be equal to the timestamp of the first sample of the adjacent segment of the reference track, but it is recommended it be in the range of the segment containing this producer reference time box.

Annex H (normative): MIME Type Registration for MPD

H.1 MPD MIME Type

H.1.1 Introduction

This Annex provides the formal MIME type registration for the MPD. It is referenced from the registry at http://www.iana.org/.

H.1.2 MIME Type and Subtype

The MIME Type and Subtype are defined as follows:

Media Type Name: application

Subtype name: dash+xml

Required parameters: none

Optional parameters: The "profiles" parameter as documented in the published specification

Encoding considerations: as for application/xml

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.

Interoperability considerations:
The specification defines a platform-independent expression of a presentation, and it is intended that wide interoperability can be achieved.

Published specification: 3GPP TS 26.247

Applications which use this media type: various

Additional information:

Magic number(s): none

File extension(s): mpd

Macintosh File Type Code(s): n/a

Object Identifier(s) or OID(s): none

Intended usage: common

Other Information/General Comment: none

Person to contact for further information:

Name: Thomas Stockhammer

Email: stockhammer@nomor.de

Author/Change controller: 3GPP

H.1.3 Parameters

H.1.3.1 The profiles parameter

Parameter name: Profiles

Parameter value: A single value, or a comma-separated list of values identifying the profiles(s) to which the file claims conformance.

Note that, per RFC 2045, some characters (including the comma used to separate multiple values) require that the entire parameter value be enclosed in quotes.

An element may include an octet that must be encoded in order to comply with RFC 2045. In this case, RFC 2231 is used: an asterisk ("*") is placed at the end of the parameter name (becoming 'profiles*' instead of 'profiles'), the parameter value usually starts with two single quote ('') characters (indicating that neither character set nor language is specified), and each octet that requires encoding is represented as a percent sign ('%') followed by two hexadecimal digits. Note that, when the RFC 2231 form is used, the percent sign, asterisk, and single quote characters have special meaning and so must themselves be encoded.

The 'profiles' parameter is an optional parameter that indicates one or more profiles to which the file claims conformance. The value is a space-delimited list of profile identifiers. 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 11).

EXAMPLE:

```
application/dash+xml;profiles='urn:3GPP:PSS:profile:XY urn:3GPP:PSS:profile:XZ'
```

H.2 Delta MPD MIME Type

H.2.1 Introduction

This Annex provides the formal MIME type registration for the delta MPD. It is referenced from the registry at http://www.iana.org/.

H.2.2 MIME Type and Subtype

The MIME Type and Subtype are defined as follows:

Media Type Name: application

Subtype name: deltadash+xml

Required parameters: none

Optional parameters: none

Encoding considerations: as for application/xml

Security considerations: The Delta 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 Delta 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.

Interoperability considerations:
The specification defines a platform-independent expression of a presentation, and it is intended that wide interoperability can be achieved.

Published specification: 3GPP TS 26.247

Applications which use this media type: various

Additional information:

Magic number(s): none

File extension(s): mpdd

Macintosh File Type Code(s): n/a

Object Identifier(s) or OID(s): none

Intended usage: common

Other Information/General Comment: none

Person to contact for further information:

Name: Thomas Stockhammer

Email: stockhammer@nomor.de

Author/Change controller: 3GPP

Annex I (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
2011-06	52	SP-110305			Version 10.0.0 approved at TSG SA#52		10.0.0

History

Document history		
V10.0.0	June 2011	Publication