INTERNATIONAL STANDARD

ISO/IEC 23009-1

Second edition
2014-05-15

AMENDMENT 3
2016-10-15

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

Part 1: Media presentation description and segment formats

AMENDMENT 3: Authentication, MPD linking, Callback Event, Period Continuity and other Extensions

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

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

AMENDEMENT 3: Authentification, liaison MPD, événement de rapel, continuité de la période et autres extensions

Reference number
ISO/IEC 23009-1:2014/Amd.3:2016(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Foreword

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

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

Amendment 3 to ISO/IEC 23009-1:2014 was prepared by ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

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

Part 1:
Media presentation description and segment formats

AMENDMENT 3: Authentication, MPD linking, Callback Event, Period Continuity and other Extensions

Page 2, Terms and definitions

Add the following definition in 3.1:

3.1.X
sub-asset
media content component (or part thereof) identified as corresponding to a part of an asset

Page 15, Table 2

Add the following identifiers in Table 2:

Scheme Identifier	Clause in this document	Informative description
urn:mpeg:dash:period-continuity:2015	5.3.2.4	Period continuity signalling
urn:mpeg:dash:period-connectivity:2015	5.3.2.4	Period connectivity signalling
urn:mpeg:dash:mpd-as-linking:2015	5.8.5.X	MPD Adaptation Set Linking scheme.
urn:mpeg:dash:sai:2015	5.8.5.Y	Sub-Asset Scheme Identifier
urn:mpeg:dash:client-authentication:2015	5.8.5.Z	Client Authentication scheme.
urn:mpeg:dash:content-authorization:2015	5.8.5.Z	Content Access Authorization scheme.
urn:mpeg:dash:event:callback:2015	5.10.4.X	DASH call back event.

Page 9, Clause 4

Add in the second paragraph of 4.3, discussing Assets:

Same asset over multiple periods may be identified by a DASH descriptor enabling DASH clients to maintain the continuity across periods' boundaries. Furthermore, sub-assets composing the same asset may also identified using a similar method. For instance, if an asset is composed of multiple video components, sub-assets enable selecting the previously selected video component after an ad insertion.

As such an asset can contain sub-assets that can be uniquely identified from one media content period to another.

Page 23, Table 4

Add at the end of Table 4:

EmptyAdaptationSet	0...N	specifies an Adaptation Set that does not contain any Representation element. The empty Adaptation Set is of the same type as a regular Adaptation Set, but shall neither contain an xlink nor may it contain any Representation element. This element shall only be present, if an Essential Descriptor is present with @schemeIDURI set to "urn:mpeg:dash:mpd-as-linking:2015". For more details, see 5.8.5.8.

Page 25, Clause 5

Add to schema:

```
  <!-- Period -->
  <xs:complexType name="PeriodType">
    <xs:sequence>
      <xs:element name="BaseURL" type="BaseURLType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="SegmentBase" type="SegmentBaseType" minOccurs="0"/>
      <xs:element name="SegmentList" type="SegmentListType" minOccurs="0"/>
      <xs:element name="SegmentTemplate" type="SegmentTemplateType" minOccurs="0"/>
      <xs:element name="AssetIdentifier" type="DescriptorType" minOccurs="0"/>
      <xs:element name="EventStream" type="EventStreamType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="AdaptationSet" type="AdaptationSetType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="Subset" type="SubsetType" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="SupplementalProperty" type="DescriptorType" minOccurs="0"/>
      <xs:element name="EmptyAdaptationSet" 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="bitstreamSwitching" type="xs:boolean" default="false"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
```

Page 25, Clause 5

Add new subclause:

5.3.2.4 Content Offering with multiple Periods

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

— to enable splicing of content, for example for ad insertion;

— to provide synchronization in segment numbering, e.g. compensate non-constant segment durations;

— to remove or add certain Representations in an Adaptation Set;

— to remove or add certain Adaptation Sets;

— to remove or add content offering on certain CDNs;

— to enable signalling of shorter segments, if produced by the encoder.

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

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

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

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

— The Adaptation Sets are associated.

— The sum of the value of the `@presentationTimeOffset` and the presentation duration of the Representations in one Adaptation Set are identical to the value of the `@presentationTimeOffset` of the associated Adaptation Set in the next Period.

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

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

— the `@value` of the descriptor matching the value of an `@id` of a Period that is contained in the MPD, and

— the value of the **`AdaptationSet`**`@id` being the same in both Periods.

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

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

Note that period continuity implies period connectivity.

For appropriate client behaviour, please refer to A.9.

Page 74, Clause 5

Add new subclause:

5.8.5.XDASH MPD Adaptation Set Linking scheme

The URN "`urn:mpeg:dash:mpd-as-linking:2015`" is defined in order to provide information that the same Adaptation Set can be found in another MPD. The scheme may be used with Essential Property Descriptors together with an **`EmptyAdaptationSet`** element or with Supplemental Property Descriptors. The value provides

— a URL to the MPD, including appropriate anchors for Periods and Adaptation Set as defined in C.4.2, and

— optionally, a timeline offset field to synchronize the data added as a white-space separated second item. If the value is not present, then the media time on the original MPD and the linked MPD are identical. If a `timeOffset` field is added, then this value expresses the difference between the media time of the Adaptation Set of the linked MPD and the media time in the originating MPD. Note that the number may be positive or negative. The `timeoffset` is in unit of `@timescale` of the linked Adaptation Sets.

The Descriptor shall only be used in combination with Adaptation Sets (regular or empty ones). If all information is contained in the Adaptation Set, then a Supplemental Descriptor or an Essential Descriptor may be used. If the Adaptation Set does neither contain a **`Representation`** element, nor an `@xlink` attribute, then an Empty Adaptation Set as well as an Essential Descriptor shall be used to indicate that the Adaptation Set is not fully described in this MPD.

NOTE This scheme may be used for many use cases. However, one specific use case is the server-based mosaic channel as described in ISO/IEC 23009-3. The use case may be fulfilled by the combination of the Spatial Relationship Description (SRD) as defined in Annex H and this scheme.

Page 74, Clause 5

Add new subclause:

5.8.5.Y Sub-Asset Scheme Identifier

In DASH MPD, sub-assets across Periods can be identified using the sub asset Scheme Identifier. This scheme is signalled using a specific **`SupplementalProperty`** descriptor at the Adaptation Set or Sub-Representation level with `@schemeIdUri` attribute set to "`urn:mpeg:dash:sai:2015`".

If two different Adaptation Sets or Sub-Representation from different Periods contain Sub-Asset Identifiers descriptors with the same `@value` attribute, then the content in the Representation(s) contained in these Adaptation Sets represent, at least, the same sub-asset.

NOTE 1 The association between sub-assets and an Adaptation Set may change between Periods. For instance, an Adaptation Set may be associated with a sub-asset of an asset in one Period but with another sub-asset of the same asset in another Period.

NOTE 2 Sub-Asset Identifier descriptor may be used by DASH clients to select Representation(s) to be processed after a Period change.

A given Adaptation Set may contain more than one Sub-Asset Identifier descriptor indicating that the Representation(s) contained in this Adaptation Set represent more than one identified part of the asset.

NOTE 3 If the value for this descriptor is not recognized, the SubAsset Identifier descriptor may still be used to understand the equivalence of sub-asset identifiers across Periods. Processing of the descriptor scheme and value by the DASH client is not essential for normal operation.

Page 74, Clause 5

Add new subclause:

5.8.5.Z Client Authentication and Content Access Authorization

When client authentication and/or content access authorization functionality is needed, DASH may be used with different schemes, such as Open Authentication Technical Committee (OATC) Online Multimedia Authorization Protocol (OMAP),[13] Open Standard for Authorization (OAuth) 2.0,[14] OASIS Security Assertion Markup Language (SAML),[15] 3GPP Generic Authentication Architecture (GAA),[16] or 3GPP Generic Bootstrapping Architecture (GBA).[17] This subclause describes generic signalling to support use of various authentication and authorization schemes.

Typical access control methods include blocking HTTP requests that do not include a security token obtained by the authorization protocol wherein the security token is validated by a CDN before downloading the requested Media Segment, or encrypting Media Segments so that playback will be restricted unless the authorization protocol provides the client with a decryption key.

For client authentication, a service may limit content delivery to authenticated clients and may use client identification information such as certificates, cookies, and embedded keys to determine subscription rights, etc. required to authorize playback of the Media Presentation. The details of such a scheme are outside the scope of this document. A client that does not support the signalled content access authorization would not be able to play the content.

NOTE This subclause does not provide any requirements on client authentication or content access authorization.

The signalling and setup of the specific scheme may be done outside the MPD level, e.g. a system applying such a scheme may only permit access to the MPD if the client is authenticated or the content access is authorized.

However, there are cases for which the MPD is provided without access control. In this case, client authentication and content access authorization methods may be signalled in the MPD using Essential Property Descriptors. The DASH client may be offered with multiple options to access the entire Media Presentation or specific parts of the Media Presentation, e.g. a specific Adaptation Set. An Essential Property descriptor may be placed at the appropriate level, and, for example, the **EssentialProperty**`@schemeIdUri` may signal the URN of the appropriate authentication or authorization method and the **EssentialProperty**`@value` attribute may carry some scheme specific information. Other signalling may be used, but the detailed signalling and semantics remain specific to a particular scheme.

There may be cases in which multiple options are provided to the client for client authentication and/or content access authorization. In this case, the **EssentialProperty**`@id` value may be used to signal functional equivalence of descriptors. In the absence of other information, the **EssentialProperty**`@id` value may contain the following URNs:

— `urn:mpeg:dash:client-authentication:2015` for client authentication;

— `urn:mpeg:dash:content-authorization:2015` for content access authorization.

In this case,

— each **EssentialProperty** descriptor with **EssentialProperty**`@id` value of `urn:mpeg:dash:client-authentication:2015` indicates a supported client authentication protocol. A client may select one of possibly multiple elements with that **EssentialProperty**`@id` value and a scheme that it recognizes based on the `@schemeIdUri` attribute; and execute that protocol using any information included in the `@value` attribute and any extension elements defined by that particular authentication scheme;

— each **EssentialProperty** descriptor with **EssentialProperty**@id value of urn:mpeg:dash:content-authorization:2015 indicates a supported content access authorization protocol. A client may select one of possibly multiple elements with that **EssentialProperty**@id value and a scheme that it recognizes based on the @schemeIdUri attribute; and execute that protocol using any information included in the @value attribute and any extension elements defined by that particular authorization scheme.

A DASH client that is successfully authenticated as an authenticated player and authorized for playback of some or all Representations or Adaptation Sets in the MPD may request and play the authorized content.

Page 74, Clause 5

Add new subclause:

5.10.4.X DASH Callback Event

5.10.4.X.1 General

DASH Callback events are indications in the content that it is expected by a DASH client to issue an HTTP GET request to a given URL and ignore the HTTP response. These event schemes are identified by the URN "urn:mpeg:dash:event:callback:2015".

A content author may use such an event for tracking play-back of specific content on a server that is not included in the media path.

The Status-Code field of the HTTP response should be 2xx, however the client is expected to entirely ignore the response.

NOTE 1 HTTP GET (as opposed to HEAD) is used in alignment with IAB VAST.[18]

The message body of the HTTP response should be as small as possible or absent.

NOTE 2 The system adopting this functionality is expected to define appropriate means for secure handling of this feature.

5.10.4.X.2 Inband event

Table 5.10.4.X.1 defines the message data and the expected actions for different @value values when the DASH callback event is signalled as an Inband Event.

Table 5.10.4.X.1 — Message data and expected actions for DASH call back inband event

value	message_data[]	Action
1	Valid HTTP/HTTPS URL	An HTTP GET request is expected to be issued to a URL contained in message_data[]. The URL shall be a NULL-terminated string. HTTP response shall either not be provided or be provided such that it can be discarded.

5.10.4.X.3 MPD event

Table 5.10.4.X.2 defines the relevant parameters for a call back event signalled in the MPD.

Table 5.10.4.X.2 — Relevant parameters for a call back event signalled in the MPD

Attribute	Value
EventStream@schemeIdUri	"urn:mpeg:dash:event:callback:2015"
EventStream@value	1
Event@messageData	HTTP-URL HTTP response is expected to be discarded without parsing.

Page 114, Annex A

Add new A.9:

A.9 Playback across Period boundaries

From a client perspective, Period boundaries may require processing that makes fully continuous playout impossible or at last practically complex. For example, the content may be offered with different codecs, different language attributes, different protection and so on. The client is expected to play the content continuously across Periods, but there may be implications in terms of implementation to provide fully continuous and seamless playout. It may be the case that at Period boundaries, the presentation engine needs to be re-initialized, for example due to changes in formats, codecs or other properties. This may result in a re-initialization delay. Such a re-initialization delay should be minimized.

If the client presents media components of a certain Adaptation Set with a specific value `foo` for the **AdaptationSet**@id in one Period, and if the following Period has assigned an identical Asset Identifier, then the client is suggested to identify an associated Period and, in the absence of other information, continue playing the content in the associated Adaptation Set, i.e. the Adaptation Set with value `foo` for the **AdaptationSet**@id.

If the client presents media components of a certain Sub-Representation in one Period, and if the following Period has assigned an identical Sub-Asset Identifier, then the client is suggested to identify an associated Period and, in the absence of other information, continue playing the content in the associated Sub-Representation.

If furthermore the Adaptation Sets are *period-continuous* or *period-connected* as defined in 5.3.2.4, i.e. the presentation times are continuous and this is signalled in the MPD, then the client is expected to seamlessly play (as defined in 4.5.1) the content across the Period boundary. Most suitably, the client may continue playing the Representation in the Adaptation Set with the same @id, but there is no guarantee that this Representation is available. In this case, the client is expected to seamlessly switch (as defined in 4.5.1) to any other Representation in the Adaptation Set. If period continuity is signalled and if continuously playing, then the client should ignore the value of the @presentationTimeOffset attribute and just continuing processing the incoming Segments. If period connectivity is signalled and if continuously playing, then the client is expected to inform the media decoder on a timeline discontinuity obeying the value of @presentationTimeOffset attribute, but it may continuing processing the incoming Segments without, for example, re-initializing the media decoder.

Page 115, Annex B

Update Annex B:

Annex B
(normative)

MPD schema

The schema of the MPD for this edition of the document is provided below.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="urn:mpeg:dash:schema:mpd:2011"
attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.
w3.org/2001/XMLSchema" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns="urn:mpeg:dash:schema:mpd:2011">

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

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

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

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

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

```
    </xs:restriction>
  </xs:simpleType>

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

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


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

  <!-- Adaptation Set -->
  <xs:complexType name="AdaptationSetType">
    <xs:complexContent>
      <xs:extension base="RepresentationBaseType">
        <xs:sequence>
          <xs:element name="Accessibility" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
          <xs:element name="Role" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
          <xs:element name="Rating" type="DescriptorType" minOccurs="0"
maxOccurs="unbounded"/>
          <xs:element name="Viewpoint" type="DescriptorType" minOccurs="0"
```

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

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

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


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

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

```
    <xs:attribute name="par" type="RatioType"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

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

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


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

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

```
    <xs:attribute name="segmentProfiles" type="xs:string"/>
    <xs:attribute name="codecs" type="xs:string"/>
    <xs:attribute name="maximumSAPPeriod" type="xs:double"/>
    <xs:attribute name="startWithSAP" type="SAPType"/>
    <xs:attribute name="maxPlayoutRate" type="xs:double"/>
    <xs:attribute name="codingDependency" type="xs:boolean"/>
    <xs:attribute name="scanType" type="VideoScanType"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

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

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

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

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

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

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

```
    <xs:attribute name="sourceURL" type="xs:anyURI"/>
    <xs:attribute name="range" type="xs:string"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

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

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

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

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

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

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

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

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

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

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

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

</xs:schema>
```

Page 123, Table C.1

Replace Table C.1 with the following:

Table C.1 — Parameters for MPD Anchors

Key	Value	Semantics
t	Time or time range in the same format as defined in W3C Media Fragments URI 1.0. See W3C Media Fragments URI 1.0 for validity rules and recommended behaviour. Optionally prefixed comma-separated pair of numbers. See C.4.4 for validity rules and recommended behaviour.	If the parameter starts from an integer, it signifies the time since the beginning of the period indicated by the period parameter. If the t parameter is not present, its default value is t=0 (i.e. start from the beginning of the Period). NOTE if period parameter is not present, the default Period is the first period of the presentation. If the parameter starts from prefix posix:, it signifies the absolute time range defined in seconds of Coordinated Universal Time (ITU-R TF.460-6). This is the number of seconds since 01-01-1970 00:00:00 UTC. Fractions of seconds may be optionally specified down to the millisecond level. The posix notation documents the absolute time in the MPD. This notation shall only be used if **MPD**@availabilityStartTime is present. This t=posix:xxx notation parameter shall not be used if a period parameter is used. In addition, at most one of the two parameters, period and t shall be present in an anchor. A special value "now" indicates the latest available segment, i.e. "live edge".
period	String	Value of a Period parameter Period@id. If period parameter is not present, the default value of the @id attribute value of the **Period** with the earliest *PeriodStart*.
track	string	Value of a single **AdaptationSet**@id
group	string	Value of a single **AdaptationSet**@group

Page 123, Annex C

Replace text in C.4.3 with:

42nd second of Period1 `my.mpd#t=42&period=Period1`

```
my.mpd#t=42&period=Period1
```

42nd second from the start of the presentation, English 5.1 audio and video

```
my.mpd#t=42&track=en51&track=vid
```

A range from 60s to 180s of the presentation

```
my.mpd#t=60,180
```

Start a live stream at Wed, 21 Jan 2015 20:04:05 GMT

```
my.mpd#t=posix:1421870645
```

Start a live stream at Wed, 21 Jan 2015 20:04:05 GMT, English 5.1 audio and video

```
my.mpd#t=posix:1421870645&track=en51&track=vid
```

A live stream range from Wed, 21 Jan 2015 20:04:05 GMT to Wed, 21 Jan 2015 23:44:33 GMT

```
my.mpd#t=posix:1421870645,1421883873
```

Play the stream from the latest available segment to Wed, 21 Jan 2015 23:44:33 GMT

```
my.mpd#t=posix:now,1421883873
```

Play the stream from the earliest available segment to Wed, 21 Jan 2015 23:44:33 GMT

```
my.mpd#t=posix:0,1421883873
```

Page 123, Annex C

Add new subclause:

C.4.4 Handling UTC parameter

The following notations are used in this subclause.

— *N* is the UTC time at the moment the URL is requested.

— *E* is the earliest available segment time in the current presentation at time *N*.

— *F* is the latest availability segment time in the current presentation.

— *S* is the start time in the `posix` anchor.

— *T* is the end time in the `posix` anchor.

If *T* is not specified, or larger than *F*, its value shall be considered to be equal to *F*.

If *S* is not specified, its value shall be considered to be equal to *E*. If *S* is "now", its value is *N*.

Page 143, Annex G

Add new subclause:

G.10 Example for Adaptation Set linking

This example shows how to describe MPD Adaptation Set Linking scheme as defined in 5.8.5.X. In this example, main video is the server-based mosaic channel as described in ISO/IEC 23009-3.

The screen position in main video and MPD linking for each mosaic video are described by SRD scheme and MPD Adaptation Set Linking scheme in the separate Adaptation Set(s).

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
  xmlns="urn:mpeg:dash:schema:mpd:2011"
  type="dynamic"
  availabilityStartTime="2015-05-30T09:30:10Z"
  minimumUpdatePeriod="PT10S"
  minBufferTime="PT1S"
  profiles="urn:mpeg:dash:profile:live:2011">

  <ProgramInformation>
    <Title>Example of a DASH Media Presentation Description using Spatial Relationships
Description to indicate tiles of a video</Title>
  </ProgramInformation>
  <Period>
    <!-- Mosaic Video -->
    <AdaptationSet segmentAlignment="true" subsegmentAlignment="true"
subsegmentStartsWithSAP="1">
```



```
      <Role schemeIdUri="urn:mpeg:dash:role:2011" value="main"/>
      <SupplementalProperty schemeIdUri="urn:mpeg:dash:srd:2014" value="0,0,0,2,2,2,2"/>
      <SupplementalProperty schemeIdUri="urn:mpeg:dash:sai:2014" value="1"/>
      <Representation id="1" mimeType="video/mp4" codecs="avc1.42c01e" width="640"
height="360" bandwidth="226597" startWithSAP="1">
        <BaseURL> full_video_small.mp4</BaseURL>
        <SegmentBase indexRangeExact="true" indexRange="837-988"/>
      </Representation>
      <Representation id="2" mimeType="video/mp4" codecs="avc1.42c01f" width="1280"
height="720" bandwidth="553833" startWithSAP="1">
        <BaseURL> full_video_hd.mp4</BaseURL>
        <SegmentBase indexRangeExact="true" indexRange="838-989"/>
      </Representation>
      <Representation id="3" mimeType="video/mp4" codecs="avc1.42c033" width="3840"
height="2160" bandwidth="1055223" startWithSAP="1">
        <BaseURL> full_video_4k.mp4</BaseURL>
        <SegmentBase indexRangeExact="true" indexRange="839-990"/>
      </Representation>
    </AdaptationSet>
    <!-- Tile 1/Service1 -->
    <AdaptationSet>
     <SupplementalProperty schemeIdUri="urn:mpeg:dash:sai:2014" value="1"/>
      <SupplementalProperty schemeIdUri="urn:mpeg:dash:srd:2014" value="0,0,0,1,1,2,2"/>
      <EssentialProperty schemeIdUri="urn:mpeg:dash:mpd-as-linking:2015" value="http://
example.com/service1/my.mpd#period=1&as=video"/>
    </AdaptationSet>
    <!—Tile /Service 2 -->
    <AdaptationSet>
     <SupplementalProperty schemeIdUri="urn:mpeg:dash:sai:2014" value="1"/>
      <SupplementalProperty schemeIdUri="urn:mpeg:dash:srd:2014" value="0,1,0,1,1"/>
      <EssentialProperty schemeIdUri="urn:mpeg:dash:mpd-as-linking:2015" value="http://
example.com/service2/my.mpd#period=1&as=video timeOffset=70000"/>
    </AdaptationSet>
    <!—Tile 3/Service 3 -->
    <AdaptationSet>
     <SupplementalProperty schemeIdUri="urn:mpeg:dash:sai:2014" value="1"/>
      <SupplementalProperty schemeIdUri="urn:mpeg:dash:srd:2014" value="0,0,1,1,1"/>
      <EssentialProperty schemeIdUri="urn:mpeg:dash:mpd-as-linking:2015" value="http://
example.com/service3/my.mpd#period=1&as=video timeOffset=100000"/>
    </AdaptationSet>
    <!—Tile 4/Service 4 -->
    <AdaptationSet>
     <SupplementalProperty schemeIdUri="urn:mpeg:dash:sai:2014" value="1"/>
      <SupplementalProperty schemeIdUri="urn:mpeg:dash:srd:2014" value="0,1,1,1,1"/>
      <EssentialProperty schemeIdUri="urn:mpeg:dash:mpd-as-linking:2015" value="http://
example.com/service4/my.mpd#period=1&as=video timeOffset=120000"/>
    </AdaptationSet>
  </Period>
</MPD>
```

Annex I

Replace Annex I with the following:

Annex I
(normative)

Flexible Insertion of URL parameters

I.1 General

This annex describes how it is possible to configure URL parameters of media segment URLs, in a similar fashion to the URL template mechanism. The core specification of DASH already allows building media segment URLs containing static URL parameters. This annex aims at providing more flexibility and dynamicity in the way URL parameters are inserted.

Parameters are *instantiated* — a name-value pair is constructed by the client. Then parameters are *output* — an instantiated key-value pair is written to either query parameters or headers, depending on the *output mode*.

The mechanism described in this annex allows several methods of instantiation:

— "Inheritance" from MPD URL parameters when the MPD is delivered over HTTP, i.e. extraction of one or more key-value pairs from the query string of the URL used to fetch MPD;

— Just-in-time server-side instantiation using the XLink mechanism;

— Header instantiation: given a header name, the parameter value is the value of the header;

— Client-side computation — specific feedback (uniquely identified by URN) is expected from the client side.

There are two output modes described by this annex:

— Query parameters: parameters are written as key-value pairs in HTTP GET requests issued by the DASH client;

— HTTP header: parameters are written as a list of key-value pairs in a DASH-specific HTTP header.

Flexible insertion of URL parameters is signalled through the use of **`EssentialProperty`** or **`SupplementalProperty`** descriptors, with `@schemeIdUri` values defined below:

— Baseline segment URL parametrization scheme identified by URN "`urn:mpeg:dash:urlparam:2014`" and described in I.2. This scheme applies only to requests for media segments.

— Extended parametrization scheme applying to requests for media segments, MPD, XLink and callback events, and is a superset of the baseline scheme. This scheme is identified by URN "`urn:mpeg:dash:urlparam:2016`" and described in I.3.

I.2 Segment URL parametrization

Flexible insertion of URL parameters is signalled through the use of **`EssentialProperty`** or **`SupplementalProperty`** descriptors, with `@schemeIdUri` equal to "`urn:mpeg:dash:urlparam:2014`".

A child element `up:`**`UrlQueryInfo`** shall be present in these descriptors, defined within the "`urn:mpeg:dash:schema:urlparam:2014`" namespace. The namespace prefix should be "`up:`".

As defined by this specification, each of these descriptors may be present at the MPD, Adaptation Set or at the Representation level. Only **`SupplementalProperty`** descriptor may be present at the Period level.

When insertion of URL parameters is required for a Period, **EssentialProperty** descriptors shall be inserted in all Adaptation Sets of that Period. At most one descriptor shall be present at each level.

When the insertion of URL parameters requires scheme-dependent computation, one or several additional **EssentialProperty** or **SupplementalProperty** descriptors shall be present. These descriptors shall carry an appropriate `@schemeIdUri` attribute referencing the scheme to be used and provide sufficient information to appropriately compute the required URL parameters (see I.2.3.2). Support of these schemes is not in the scope of this specification.

I.2.1 URL Query Information

I.2.1.1 Overview

The **UrlQueryInfo** element describes how to build a URL query string, which is used in the media segments URLs building process.

This query string can come from one of the three sources below:

— the URL of the MPD when the `@useMPDUrlQuery` is set;

— the `@queryString` attribute when present;

— the `@queryString` attribute, after any XLink resolution in case `@xlink:href` is present.

The `@queryTemplate` attribute describes which URL parameters contained in the query string are used in the media segment URL building process, as well as the order of these parameters.

I.2.1.2 Semantics

Table I.1 — Semantics of `UrlQueryInfo` element

Element or Attribute Name	Use	Description
`UrlQueryInfo`		provides URL query string information
`@queryTemplate`	O (string)	provides URL parameters template information This string shall contain one or more `$<ParamIdentifier>$` template identifiers, as listed in Table I.2. These template identifiers are to be replaced to build a query string (see I.2.3). If $<ParamIdentifier>$ is not in Table I.2, it will be replaced with an empty string. If the template has an opening $ without a matching closing $, the result is undefined, and the client will act as if it did not understand the **`EssentialProperty`**'s scheme. When selection of URL parameters is enabled through the use of `$query:param$` template identifiers, URL parameters shall be defined as name=value pairs separated by &, as defined by W3C HTML 4.01 Specification (section on Forms#Form submission).
`@useMPDUrlQuery`	O (bool) default: false	indicates that the URL parameters of the MPD URL are used in the construction of media segment URLs. This attribute may only be present when the MPD is delivered over HTTP, and defaults to "false" when the MPD is not delivered over HTTP. If `@queryString` is present and the value of this attribute is "true", concatenation of MPD parameter string and `@queryString` shall be used for constructing the query string of media segment URLs NOTE simple parameter signalling may be used (`"a=X&b=Y"`), as well as scheme-dependent signalling (`"a=$urn:XYZ$&b=$urn:ABC$"`). When scheme-dependent signalling is used, the scheme shall be inserted between two enclosing $ characters. See I.2.3.2 for further details. When **`UrlQueryInfo`** element is present at more than ole level of the hierarchical data model (e.g., MPD and Period), there shall be at most one **`UrlQueryInfo`** element for which `@useMPDUrlQuery` is true within this hierarchy. See I.2.3.2 for further details.
`@queryString`	O (string)	provides a query string to be used in the construction of media segment URLs. NOTE simple parameter signalling may be used (`"a=X&b=Y"`), as well as scheme-dependent signalling (`"a=$urn:XYZ$&b=$urn:ABC$"`). When scheme-dependent signalling is used, the scheme shall be inserted between two enclosing $ characters. See I.2.3.2 for further details.
`@xlink:href`	O	specifies a reference to a remote **`UrlQueryInfo`** element
`@xlink:actuate`	OD default: `onRequest`	specifies the processing instructions, which can be either "`onLoad`" or "`onRequest`". This attribute shall not be present if the `@xlink:href` attribute is not present.
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 I.2 — Parameter identifiers

$<ParamIdentifier>$	Substitution parameter
$$	Is an escape sequence, i.e. "$$" is replaced with a single "$"
$*querypart*$	This identifier is substituted with the query part of the computed query string (referred to as *initialQueryString* in I.2.3). This identifier shall not appear more than once in the template string, and shall not be mixed with other identifiers (e.g. $*query:<param>*$ below). The query part starts after the "?" sign and lasts until the "#" sign
$*query:<param>*$	This identifier is substituted with the value of the <param> parameter if this parameter is present in the query part of the computed query string (referred to as *initialQueryString* in I.2.3). If same <param> value appears more than once in the query string, the last value will be used. If <param> is not present, the empty string will be used. When this parameter selection is used, URL parameters shall be inserted in the query part of the URL as name=value pairs separated by the "&" symbol, according to W3C HTML 4.01 specification (section on Forms#Form submission)

I.2.1.3 XML Syntax

```
<!-- Url query information -->
<xs:complexType name="UrlQueryInfoType">
  <xs:attribute name="queryTemplate" type="xs:string"/>
  <xs:attribute name="useMPDUrlQuery" type="xs:boolean"/>
  <xs:attribute name="queryString" type="xs:string"/>
  <xs:attribute name="xlink:href"/>
  <xs:attribute name="xlink:actuate" default="onRequest"/>
</xs:complexType>

<xs:element name="UrlQueryInfo" type="UrlQueryInfoType"/>
```

I.2.3 Modified template-based segment URL construction, according to `UrlQueryInfo` element

When signalized through an appropriate descriptor, containing a **`UrlQueryInfo`** element, the following media segment URL building process is performed.

The process is defined in the following steps.

a) If this **`UrlQueryInfo`** element is a remote element, it is dereferenced. This process is defined in 5.5.

b) Initial query string (referred to as *initialQueryString*) is derived. This process is described in I.2.3.1.

c) Final query string (referred to as *finalQueryString*) is computed, according to `@queryTemplate` and *initialQueryString*. This process is described in I.2.3.2.

d) Final query string (*finalQueryString*) is processed to build media segment URLs. This process is described in I.2.3.3.

I.2.3.1 Computation of an initial query string (*initialQueryString*)

The initial query string *initialQueryString* is constructed by concatenating the query strings, if present and available, coming from the MPD URL (if `@useMPDUrlQuery` is set to "`true`") and `@queryString` (possibly after dereferencing). If `@useMPDUrlQuery` is set to "`true`" and `@queryString` if present, *initialQueryString* shall be a concatenation of query string from the MPD and `@queryString` string, in this order.

When multiple strings are appended together, an "&" symbol shall be inserted at the start of the second and following strings to be appended.

I.2.3.2 Computation of a final query string (*finalQueryString*)

A final query string *finalQueryString* is then computed by substituting URL parameters templates present in `@queryTemplate` by their values provided in *initialQueryString*, according to Table I.2.

When two or more occurrences of URL query descriptors exist within an MPD, the *finalQueryString* string used at the Representation level is a concatenation of the corresponding URL query strings of the occurrences in their orders of appearance in the MPD hierarchy. The query coming from the MPD URL is appended first. Thus, for each representation, the concatenation shall be computed as a concatenation of Representation-level query string with (in this order) AdaptationSet-level string, Period-level query string and, lastly, MPD-level query string.

Simple parameter signalling may be used (`@queryString = "a=X&b=Y"`), as well as scheme-dependent signalling (`@queryString = "a=$urn:XYZ$&b=$urn:ABC$"`). In the latter case, the client needs to be aware of the provided schemes and has to compute appropriate values for them. Further, in this case, additional **`EssentialProperty`** or **`SupplementalProperty`** descriptors, at the same level as the query descriptor, shall be present to reflect that scheme-dependent signalling is used and required to be supported by the client. These descriptors shall have the `@schemeIdUri` attribute set to the same value as used in the URL parameter insertion description (i.e. to "`urn:XYZ`" or "`urn:ABC`" in the above example). Support of these specific schemes is out of the scope of this specification.

A straightforward implementation of the process would do the following.

a) Create a parameter table out of concatenated `initialQueryString`. The latter is an '&'-separated list of `<param>=<value>` strings, and each `<param>=<value>` string is converted into a single entry in the parameter table. For example, for a string "`param0=42`" in `initialQueryString`, we will have `parameter["param0"] = 42`. Note that if a string "`param0=42`" is followed by a string "`param0=43`" later in `initialQueryString`, then `parameter["param0"] = 43`. If `<param>` is a URN, then `<value>` is computed by the client (and is an empty string otherwise).

b) Search for the "`$query:`" substring in the `@queryTemplate` attribute. For each appearance of this substring, the characters till the first '$' character are the parameter name (`<param>` in our notation). Substitute the complete `$query<param>$` string (including the opening and the closing '$' characters) with `parameter[<param>]`. For example, given `@queryTemplate="p0=$query:param0"`, and given `parameter["param0"] = 43`, the result would be `finalQueryString="p0=43"`.

I.2.3.3 Modified media segment URLs building process

The computed final query string *finalQueryString* is then concatenated to media Segment URLs.

If the original media segment URL does not contain any query (as defined in RFC 3986), a "?" character shall be inserted accordingly between the original media segment URL and the *finalQueryString* when performing this concatenation.

If the original media segment URL already contains a query (as defined in RFC 3986), an "&" character shall be inserted between the original media segment URL and the *finalQueryString* when performing this concatenation.

When Annex E is used together with flexible insertion of URL query parameters, processing of URL query parameters descriptors shall occur first, followed by Annex E byte range requests building process.

I.2.4 Examples

I.2.4.1 Example 1

Here, the intent is to re-use the URL parameters of the MPD URL in the media segments URLs.

Assuming DASH MPD is accessible through http://www.example.com/dash/urlparam1.mpd?token=1234&ip=1.2.3.4

```
<AdaptationSet mimeType="video/mp4" segmentAlignment="true" startWith-
SAP="1" maxWidth="1280" maxHeight="720" maxFrameRate="25" par="16:9">
  <EssentialProperty schemeIdUri="urn:mpeg:dash:urlparam:2014"
xmlns:up="urn:mpeg:dash:schema:urlparam:2014">
    <up:UrlQueryInfo queryTemplate="$querypart$" useMPDUrlQuery="true"/>
  </EssentialProperty>
  <SegmentTemplate duration="2" startNumber="1" media="video_$Num-
ber$_$Bandwidth$bps.mp4">
  </SegmentTemplate>
    <Representation id="v0" codecs="avc3.4d401f" width="1280"
height="720" frameRate="25" sar="1:1" bandwidth="3000000"/>
    <Representation id="v1" codecs="avc3.4d401f" width="640" height="360"
frameRate="25" sar="1:1" bandwidth="1500000"/>
</AdaptationSet>
```

a) Computation of an initial query string:

initialQueryString = "token=1234&ip=1.2.3.4"

b) Computation of a final query:

finalQueryString = "token=1234&ip=1.2.3.4"

c) Modified media segment URLs building process:

http://www.example.com/dash/video_1_3000000bps.mp4?token=1234&ip=1.2.3.4

http://www.example.com/dash/video_2_3000000bps.mp4?token=1234&ip=1.2.3.4

http://www.example.com/dash/video_3_3000000bps.mp4?token=1234&ip=1.2.3.4

I.2.4.2 Example 2

Here, the intent is to dynamically compute some URL parameters before adding them to the media segments URLs.

Assuming DASH MPD is accessible through http://www.example.com/dash/urlparam2.mpd and that http://www.example.com/dash/xlinked.mpd contains the following **`UrlQueryInfo`** element:

```
<up:UrlQueryInfo xmlns:up="urn:mpeg:dash:schema:urlparam:2014" queryTem-
plate="$querypart$" queryString="param=justintimecomputedvalue"/>
```

```
<AdaptationSet mimeType="video/mp4" segmentAlignment="true" startWith-
SAP="1" maxWidth="1280" maxHeight="720" maxFrameRate="25" par="16:9">
  <EssentialProperty schemeIdUri="urn:mpeg:dash:urlparam:2014"
xmlns:up="urn:mpeg:dash:schema:urlparam:2014">
    <up:UrlQueryInfo xlink:href="http://www.example.com/dash/xlinked.mpd"
xlink:actuate="onRequest"/>
  </EssentialProperty>
  <SegmentTemplate duration="2" startNumber="1" media="video_$Num-
ber$_$Bandwidth$bps.mp4">
  </SegmentTemplate>
    <Representation id="v0" codecs="avc3.4d401f" width="1280"
height="720" frameRate="25" sar="1:1" bandwidth="3000000"/>
    <Representation id="v1" codecs="avc3.4d401f" width="640" height="360"
frameRate="25" sar="1:1" bandwidth="1500000"/>
</AdaptationSet>
```

a) Computation of an initial query string (computed on request, according to xlink:actuate):

initialQueryString = "param=justintimecomputedvalue"

b) Computation of an final query string:

finalQueryString = "param=justintimecomputedvalue"

c) Modified media segment URLs building process:

http://www.example.com/dash/video_1_3000000bps.mp4?param=justintimecomputedvalue

http://www.example.com/dash/video_2_3000000bps.mp4?param=justintimecomputedvalue

http://www.example.com/dash/video_3_3000000bps.mp4?param=justintimecomputedvalue

I.2.4.3 Example 3

Here the intent is asking the client for some feedback through URL parameters (GPS location here).

Assuming DASH MPD is accessible through http://www.example.com/dash/urlparam3.mpd?pd=$urn:example:gps$ and that "`urn:example:gps`" informs the client that it should insert its GPS coordinates.