INTERNATIONAL STANDARD ISO/IEC 23009-1:2014
TECHNICAL CORRIGENDUM 1
Published 2015-04-01

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION
INTERNATIONAL ELECTROTECHNICAL COMMISSION • МЕЖДУНАРОДНАЯ ЭЛЕКТРОТЕХНИЧЕСКАЯ КОМИССИЯ • COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

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

Part 1:
Media presentation description and segment formats

TECHNICAL CORRIGENDUM 1

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

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

RECTIFICATIF TECHNIQUE 1

Technical Corrigendum 1 to ISO/IEC 23009-1:2013 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information.*

Replace

4.5.1 Switching and Random Access Support

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

ICS 35.040 **Ref. No. ISO/IEC 23009-1:2014/Cor.1:2015(E)**

Published in Switzerland

standard provides means for providing suitable switching opportunities and in addition provides abilities to signal the position and media time of the switching opportunities.
For this purpose this subsection defines three relevant concepts to support seamless switching, namely

- Media stream access points in 4.5.2 to signal positions where to easily switch to a Representation, and in addition where to suitable access a Representation at start-up or seek.

- Non-overlapping Segments and Subsegments in 4.5.3 to signal that at the signalled stream access points, no overlap decoding of Representations is necessary in order to provide a continuous switch

- Bitstream concatenation in 4.5.4 to signal that the concatenation of two Representations at a switch point results in a conforming bitstream

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

with

4.5.1 Switching and Random Access Support

The formats defined in this part of the standard are designed for providing good user experience even in case the access bandwidth between the DASH segment delivery function or the cache varies. A key functionality is the ability that the DASH client can seamlessly switch across different Representations of the same media component without severely impacting the user experience.
Assume two Representations A and B. A switch from Representations A to Representation B at media time t is considered seamless, if the result of the presentation a fter this switch is applied is identical as if Representation A is decoded from the beginning and presented up to time t and Representation B is decoded from the beginning and presented from time t onwards.
Media Presentations may provide different Representations in one Adaptation Set representing the same media component. If such Representations are properly time-aligned (as expected by the Media Presentation), then DASH clients may apply seamless switching across different Representations provided in one Adaptation Set at any time t to obtain a perceptually continuous experience.
However, in practical implementations, the operation of seamless switching may be complex, as switching at time t may require parallel download and decoding of two Representations. Therefore, providing suitable switching opportunities in regular time intervals simplifies client implementations. This part of ISO/IEC 23009-1 provides means for providing suitable switching opportunities and in addition provides abilities to signal the position and media time of the switching opportunities.
For this purpose this subsection defines three relevant concepts to support seamless switching:

- Media stream access points in 4.5.2 to signal positions where to easily switch to a Representation, and in addition where to suitable access a Representation at start-up or seek.

- Non-overlapping Segments and Subsegments in 4.5.3 to signal that at the signalled stream access points, no overlap decoding of Representations is necessary in order to provide a continuous switch.

- Bitstream concatenation in 4.5.4 to signal that the concatenation of two Representations at a switch point results in a conforming bitstream.

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

In 4.7 Schemes, Table 2 replace

urn:mpeg:dash:23003:3:audio_channel_configuration:2011	5.8.5.4	channel configuration.
urn:mpeg:dash:outputChannelPositionList:2012	5.8.5.4	a list of output channel position to signal individual speaker positions as defined in ISO/IEC 23001-8.

With

urn:mpeg:dash:23003:3:audio_channel_configuration:2011	5.8.5.4	channel configuration. **NOTE**: legacy format for backward-compatibility, it is recommended to use the signaling as defined in ISO/IEC 23001-8 instead.
urn:mpeg:dash:outputChannelPositionList:2012	5.8.5.4	a list of output channel position to signal individual speaker positions as defined in ISO/IEC 23001-8. **NOTE**: legacy format for backward-compatibility, it is recommended to use the signalling as defined in ISO/IEC 23001-8 instead.

In 5.2.3.2 Elements and Attributes added in this Revision replace

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

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

with

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

- **MPD**@publishTime
- **MPD.EssentialProperty**
- **MPD.SupplementalProperty**
- **Period.AssetIdentifier**
- **Period.EventStream**
- **Period.SupplementalProperty**
- **RepresentationBase.InbandEventStream**
- **SegmentBase**@availabilityTimeOffset
- **SegmentBase**@availabilityTimeComplete
- **BaseURL**@availabilityTimeOffset
- **BaseURL**@availabilityTimeComplete
- **Subset**@id
- **SegmentTimeline.S**@n

In 5.3.1.2 Semantics, Table 3 add at the end

`EssentialProperty`	0 … N	specifies information about the containing element that is considered essential by the Media Presentation author for processing the containing element. For details see 5.8.4.8.
`SupplementalProperty`	0 … N	specifies supplemental information about the containing element that may be used by the DASH client optimizing the processing. For details see 5.8.4.9.

In 5.3.1.3 XML syntax replace

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

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

with

```
  <!-- 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: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>
```

In 5.3.2.1 Overview replace

A Media Presentation consists of one or more Periods. A Period is defined by a **Period** element in the **MPD** element.
The type of the Period, either a regular Period or an Early Available Period, as well as the *PeriodStart* time of a regular Period is determined as follows:

- If the attribute `@start` is present in the **Period**, then the Period is a regular Period and the *PeriodStart* is equal to the value of this attribute.

- If the `@start` attribute is absent, but the previous **Period** element contains a `@duration` attribute then this new Period is also a regular Period. The start time of the new Period *PeriodStart* is the sum of the start time of the previous Period *PeriodStart* and the value of the attribute `@duration` of the previous Period.

- If (i) `@start` attribute is absent, and (ii) the **Period** element is the first in the MPD, and (iii) the **MPD**`@type` is `'static'`, then the *PeriodStart* time shall be set to zero.

- If (i) `@start` attribute is absent, and (ii) the previous **Period** element does not contains a `@duration` attribute or the **Period** element is the first in the MPD, and (iii) the **MPD**`@type` is `'dynamic'`, then this Period is an Early Available Period (see below for details).

For any regular Period the following holds: *PeriodStart* reflects the actual time that should elapse after playing the media of all prior Periods in this Media Presentation relative to the *PeriodStart* time of the first Period in the Media Presentation. The Period extends until the *PeriodStart* of the next Period, or until the end of the Media Presentation in the case of the last Period. More specifically, the difference between the *PeriodStart* time of a Period and either the *PeriodStart* time of the following Period, if this is not the last Period, or the value of the **MPD**`@mediaPresentationDuration` if this is the last one, is the presentation duration in Media Presentation time of the media content represented by the Representations in this Period.
Early Available Periods may be used to advertise initialization of other non-media data before the media data itself is available. **Period** elements documenting early available Periods shall not occur before any **Period** element documenting a regular Period. For Early Available Periods, any resources that are announced in such a **Period** element shall be available. Such a **Period** element shall not contain URLs to Media Segments. The data contained in such a **Period** element does not represent a Period in the Media Presentation. Only when the *PeriodStart* time becomes known through an update of the MPD, such a **Period** element represents a regular Period. However, an update of the MPD may even remove a **Period** element representing an Early Available Period in later updates of the MPD as long as no *PeriodStart* time is associated with the Period.

with

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

- Regular Period,

- Early Available Period,

- Early Terminated Period.

The type of the Period, as well as the *PeriodStart* time of a regular Period or early terminated Period is determined as follows:

- If the attribute `@start` is present in the **Period**, then the Period is a regular Period or an early terminated Period and the *PeriodStart* is equal to the value of this attribute.

- If the `@start` attribute is absent, but the previous **Period** element contains a `@duration` attribute then this new Period is also a regular Period or an early terminated Period. The start time of the new Period *PeriodStart* is the sum of the start time of the previous Period *PeriodStart* and the value of the attribute `@duration` of the previous Period.

— If (i) `@start` attribute is absent, and (ii) the **`Period`** element is the first in the MPD, and (iii) the **`MPD`**`@type` is `'static'`, then the *PeriodStart* time shall be set to zero.

— If (i) `@start` attribute is absent, and (ii) the previous **`Period`** element does not contain a `@duration` attribute or the **`Period`** element is the first in the MPD, and (iii) the **`MPD`**`@type` is `'dynamic'`, then this Period is an Early Available Period (see below for details).

— If (i) `@duration` attribute is present, and (ii) the next **`Period`** element contains a `@start` attribute or the `@minimumUpdatePeriod` is present, then this Period is an Early Terminated Period (see below for details).

For any regular Period and early terminated Period the following holds: *PeriodStart* reflects the actual time that should elapse after playing the media of all prior Periods in this Media Presentation relative to the *PeriodStart* time of the first Period in the Media Presentation. The Period extends until the *PeriodStart* of the next Period, or until the end of the Media Presentation in the case of the last Period. For regular Periods, the difference between the *PeriodStart* time of a Period and either the *PeriodStart* time of the following Period, if this is not the last Period, or the value of the **`MPD`**`@mediaPresentationDuration` if this is the last one, is the presentation duration in Media Presentation time of the media content represented by the Representations in this Period. For Early Terminated Periods, the value of the **`Period`**`@duration` is the presentation duration in Media Presentation time of the media content represented by the Representations in this Period.
Early Available Periods may be used to advertise initialization of other non-media data before the media data itself is available. **`Period`** elements documenting early available Periods shall not occur before any **`Period`** element documenting a regular Period. For Early Available Periods, any resources that are announced in such a **`Period`** element shall be available. The data contained in such a **`Period`** element does not represent a Period in the Media Presentation. Only when the *PeriodStart* time becomes known through an update of the MPD, such a **`Period`** element represents a regular Period. However, an update of the MPD may even remove a **`Period`** element representing an Early Available Period in later updates of the MPD as long as no *PeriodStart* time is associated with the Period.

In 5.3.2.2 Semantics replace

Table 4 — Semantics of `Period` element

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

Element or Attribute Name	Use	Description
AssetIdentifier	0...1	specifies that this Period belongs to a certain asset. For more details see 5.8.5.7.
EventStream	0...N	specifies an event stream. For more details see 5.10.2.
AdaptationSet	0...N	specifies an Adaptation Set. At least one Adaptation Set shall be present in each Period unless the value of the @duration attribute of the Period is set to zero. For more details see 5.3.3.
Subset	0...N	specifies a Subset. For more details see 5.3.8.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Note that the conditions only holds without using xlink:href. If linking is used, then all attributes are "optional" and <minOccurs=0> Elements are **bold**; attributes are non-bold and preceded with an @.		

with

Table 4 — Semantics of Period element

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

Element or Attribute Name	Use	Description
		set to 'false' for any Adaptation Set in this Period.
BaseURL	0…N	specifies a base URL that can be used for reference resolution and alternative URL selection. For more details refer to the description in 5.6.
SegmentBase	0...1	specifies default Segment Base information. Information in this element is overridden by information in **AdaptationSet.SegmentBase** and **Representation.SegmentBase**, if present. For more details see 5.3.9.
SegmentList	0...1	specifies default Segment List information. Information in this element is overridden by information in **AdaptationSet.SegmentList** and **Representation.SegmentList**, if present. For more details see 5.3.9.
SegmentTemplate	0...1	specifies default Segment Template information. Information in this element is overridden by information in **AdaptationSet.SegmentTemplate** and **Representation.SegmentTemplate**, if present. For more details see 5.3.9.
AssetIdentifier	0...1	specifies that this Period belongs to a certain asset. For more details see 5.8.5.7.
EventStream	0...N	specifies an event stream. For more details see 5.10.2.
AdaptationSet	0...N	specifies an Adaptation Set. At least one Adaptation Set shall be present in each Period unless the value of the @duration attribute of the Period is set to zero. For more details see 5.3.3.
Subset	0...N	specifies a Subset. For more details see 5.3.8.
SupplementalProperty	0...N	specifies supplemental information about the containing element that may be used by the DASH client optimizing the processing. For details see 5.8.4.9..

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 @.

In 5.3.2.3 XML syntax replace

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

with

```
  <!-- 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: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>
```

In 5.3.3.1 Overview replace

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

with

5.3.3.1 Overview

Each Period consists of one or more Adaptation Sets. An Adaptation Set is described by an **`AdaptationSet`** element. **`AdaptationSet`** elements are contained in a **`Period`** element.
An Adaptation Set contains alternate Representations, i.e. only one Representation within an Adaptation Set is expected to be presented at a time. All Representations contained in one Adaptation Set represent the same media content components and therefore contain media streams that are considered to be perceptually equivalent. The Adaptation Set and the contained Representations shall be prepared and contain sufficient information such that seamless switching (as defined in 4.5.1) across different Representations in one Adaptation Set is enabled. If an Adaptation Set is expected to be consumed by DASH clients with restrictions in terms of switching, then the Media Presentation author should provide sufficient means to enable seamless switching under these restrictions.

In 5.3.5.1 Overview add after the 1st Note

NOTE In the case a dependent Representation X depends on at least two complementary Representations Yi, that are also dependent Representations depending, directly or indirectly, on the same complementary Representation Z, then the concatenation as defined above results in a sequence conforming to the media format if the Segments of Representation Z are concatenated only the first time encountered following the order in the `@dependencyId` attributes starting from X. If following recursively the dependencies indicated in the `@dependencyId` of Representations, certain Representations may be encountered more than once. In this case the corresponding Segments or Subsegments are expected to only be concatenated once, namely when first encountered.

In 5.3.7.3 XML syntax replace

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

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

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

with

```
  <!-- 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>
```

In 5.3.9.1 General, 6th paragraph replace

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

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

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

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

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

with

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

— one **SegmentList** element - for syntax and semantics refer to 5.3.9.3,

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

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

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

In 5.3.9.3.1 Overview, 1st parapgraph replace

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

With

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

In 5.3.9.5.3 Media Segment information, 1st paragraph replace

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

with

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

In 5.3.9.5.3 Media Segment information, 4th and 5th paragraph replace

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

- if **SegmentTemplate** element is present the Template-based Segment URL construction in 5.3.9.4.4 shall be applied as follows
  - If the Representation contains or inherits a **SegmentTemplate** element with *$Number$* then the URL of the Media Segment at position *k* in the Representation is determined by replacing the $*Number*$ identifier by (*k*-1) + @startNumber.
  - If the Representation contains or inherits a **SegmentTemplate** element with *$Time$* then the URL of the media segment at position *k* is determined by replacing the $*Time*$ identifier by the time address associated to this Segment. The time address is determined as follows:
    - if the @duration attribute is present, then the time address is determined by replacing the $*Time*$ identifier with ((*k*-1) + (k_{Start}-1))* @duration with k_{Start} the value of the @startNumber attribute, if present, or 1 otherwise. Further, the media time of the Segment shall be accurately expressed by the MPD information in the following sense:
      - the value ((*k*-1) + (k_{Start}-1))* @duration is identical to the earliest presentation time in the segment.
      - the duration of the Segment in media presentation time shall be identical to the value of the @duration attribute.
    - if the **SegmentTimeline** element is present, then the time address shall be determined by replacing the $*Time*$ identifier with the earliest presentation time of the *k*-th segment as documented in the Segment timeline in 5.3.9.6.
- if one or more **SegmentList** elements are present they contain itself a list of **SegmentURL** elements for a consecutive list of Media Segment URLs. The number of the first Segment in the list within this Period is determined by the value of the **SegmentList**@startNumber attribute, if present, or it is 1 in case this attribute is not present. The sequence of multiple **SegmentList** elements within a Representation shall result in Media Segment List with consecutive numbers.
- none of the above: In this case only a single Media Segment shall be present with the URL provided by a **BaseURL** element and the **SegmentBase** element may be present.

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

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

with

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

- if **SegmentTemplate** element is present the Template-based Segment URL construction in 5.3.9.4.4 shall be applied as follows
  - If the Representation contains or inherits a **SegmentTemplate** element with *$Number$* then the URL of the Media Segment at position *k* in the Representation is determined by replacing the $*Number*$ identifier by (*k*-1) + `@startNumber`.
  - If the Representation contains or inherits a **SegmentTemplate** element with *$Time$* then the URL of the media segment at position *k* is determined by replacing the $*Time*$ identifier by the time address associated to this Segment. The time address is determined as follows:
    - if the Representation contains or inherits a **SegmentTemplate** element with *$Time$* then the URL of the media segment at position *k* is determined by replacing the $*Time*$ identifier by MPD start time of this segment, as described below.
- if one or more **SegmentList** elements are present they contain itself a list of **SegmentURL** elements for a consecutive list of Media Segment URLs. The number of the first Segment in the list within this Period is determined by the value of the **SegmentList**`@startNumber` attribute, if present, or it is 1 in case this attribute is not present. The sequence of multiple **SegmentList** elements within a Representation shall result in Media Segment List with consecutive numbers.
- none of the above: In this case only a single Media Segment shall be present with the URL provided by a **BaseURL** element and the **SegmentBase** element may be present.

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

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

In 5.3.9.6.1 General, 7th paragraph replace

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

- at least one Segment Index ('sidx') box shall be present
- the values of the **SegmentTimeline** shall describe accurate timing of each Media Segment. Specifically, these values shall reflect the information provided in the Segment index ('sidx') box, i.e.
  - the value of @timescale shall be identical to the value of the timescale field in the first 'sidx' box,
  - the value of **S**@t shall be identical to the value of the earliest_presentation_time in the first 'sidx' box of the Media Segment described in **S**,
  - the value of **S**@d shall be identical to sum of the values of all Subsegment_duration fields in the first 'sidx' box of the Media Segment described in **S**.
- The Segment URL for a Media Segment is obtained by replacing the $Time$ identifier by the earliest presentation time obtained from the **SegmentTimeline**.

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

with

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

- If a Segment Index ('sidx') box is present, then the values of the **SegmentTimeline** shall describe accurate timing of each Media Segment. Specifically, these values shall reflect the information provided in the Segment index ('sidx') box, i.e.
  - the value of @timescale shall be identical to the value of the timescale field in the first 'sidx' box,
  - the value of **S**@t shall be identical to the value of the earliest_presentation_time in the first 'sidx' box of the Media Segment described in **S**,
  - the value of **S**@d shall be identical to sum of the values of all Subsegment_duration fields in the first 'sidx' box of the Media Segment described in **S**.
- If a Segment Index ('sidx') box is not present, then the derivation of the earliest presentation time shall be based on the media internal data. The details depend on the segment format in use and further restriction on the segment format may apply.
- The Segment URL for a Media Segment is obtained by replacing the $Time$ identifier by the earliest presentation time obtained from the **SegmentTimeline**.

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

In 5.3.9.6.2 Semantics, Table 17 add after 4th row

@n	O	specifies the Segment number of the first Segment in the series. The value of this attribute must be at least one greater than the number of previous **S** elements plus the `@startNumber` attribute value, if present. If the value of @n is greater than one plus the previously calculated Segment number, it expresses that one or more prior Segments in the timeline are unavailable.

In 5.3.9.6.3 XML syntax replace

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

with

```
<!-- 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:int" use="optional" default="0"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
      </xs:complexType>
    </xs:element>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

In 5.8.4.7 Audio channel configuration, replace

For the element **`AudioChannelConfiguration`** the `@schemeIdUri` attribute is used to identify the audio channel configuration scheme employed.
Multiple **`AudioChannelConfiguration`** elements may be present indicating that the Representation supports multiple audio channel configurations. For example, it may describe a Representation that includes MPEG Surround audio supporting stereo and multichannel.

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

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

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

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

with

For the element **`AudioChannelConfiguration`** the `@schemeIdUri` attribute is used to identify the audio channel configuration scheme employed.
Multiple **`AudioChannelConfiguration`** elements may be present indicating that the Representation supports multiple audio channel configurations. For example, it may describe a Representation that includes MPEG Surround audio supporting stereo and multichannel.

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

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

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

In 5.8.4.8 Essential Property Descriptor add after 3rd paragraph

If one or more **`EssentialProperty`** elements sharing the same `@id` appear at the MPD level, this means that successful processing of at least one of these descriptors is essential to properly access and/or present the content described by this MPD.

NOTE In the case when none of the **`EssentialProperty`** elements sharing the same `@id` can be successfully processed, the DASH client is expected to terminate the media presentation.

In 5.8.5.4 Audio channel configuration schemes replace

The following defines a URI that identifies channel configuration signalling for Representations that contain an audio component. The URI "urn:mpeg:dash:23003:3:audio_channel_configuration:2011" is defined to indicate the channel configuration as defined by Table 68 (Channel Configurations, meaning of channelConfigurationIndex, mapping of channel elements to loudspeaker positions') of ISO/IEC 23003-3. The @value shall be the value as defined for OutputChannelPosition in ISO/IEC 23001-8.
The URN "urn:mpeg:dash:outputChannelPositionList:2012" defines a list of output channel positions to signal individual speaker positions. The @value shall be a space-delimited list of values as defined of the OutputChannelPosition as defined in ISO/IEC 23001-8. For example, the @value for the 7.1 channel configuration 2 high as 2/0/0, 5 mid as 3/0/2 and 0.1 low, where a/b/c indicates speaker count in front, side and back, respectively and 0.1 indicates a subwoofer channel), is "2 0 1 4 5 3 17 1".

with

The following defines a URI that identifies channel configuration signalling for Representations that contain an audio component. The URI "urn:mpeg:dash:23003:3:audio_channel_configuration:2011" is defined to indicate the channel configuration as defined by Table 68 (Channel Configurations, meaning of channelConfigurationIndex, mapping of channel elements to loudspeaker positions') of ISO/IEC 23003-3.
The URN "urn:mpeg:dash:outputChannelPositionList:2012" defines a list of output channel positions to signal individual speaker positions. The @value shall be a space-delimited list of values as defined of the OutputChannelPosition as defined in ISO/IEC 23001-8. For example, the @value for the 7.1 channel configuration 2 high as 2/0/0, 5 mid as 3/0/2 and 0.1 low, where a/b/c indicates speaker count in front, side and back, respectively and 0.1 indicates a subwoofer channel), is "2 0 1 4 5 3 17 18".

In 5.10.2.2 Semantics, Table 25 replace

Table 25 — Event Semantics

Element or Attribute Name	Use	Description
Event		specifies an event and contains the message of the event, formatted as a string. The content of this element depends on the event scheme.
@presentationTime	OD default: 0	specifies the presentation time of the event relative to the start of the Period. The value of the presentation time in seconds is the division of the value of this attribute and the value of the @timescale attribute. If not present, the value of the presentation time is 0.
@duration	O	specifies the presentation duration of the event. The value of the duration in seconds is the division of the value of this attribute and the value of the @timescale attribute. If not present, the value of the duration is unknown.
@id	O	specifies an identifier for this instance of the event. Events with equivalent content and attribute values in the **Event** element shall have the same value for this attribute.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>...<maxOccurs> (N=unbounded) Elements are **bold**; attributes are non-bold and preceded with an @.		

with

Table 25 — Event Semantics

Element or Attribute Name	Use	Description
Event		specifies an event and contains the message of the event, formatted as a string. The content of this element depends on the event scheme.
@presentationTime	OD default: 0	specifies the presentation time of the event relative to the start of the Period. The value of the presentation time in seconds is the division of the value of this attribute and the value of the @timescale attribute. If not present, the value of the presentation time is 0.
@duration	O	specifies the presentation duration of the event. The value of the duration in seconds is the division of the value of this attribute and the value of the @timescale attribute. If not present, the value of the duration is unknown.
@id	O	specifies an identifier for this instance of the event. Events with equivalent content and attribute values in the **Event** element shall have the same value for this attribute. The scope of the @id for each Event is with the same @schemeIdURI and @value pair.
@messageData	O	specifies the value for the event stream element. The value

		space and semantics must be defined by the owner of the scheme identified in the `@schemeIdUri` attribute. NOTE: this attribute is an alternative to specifying a complete XML element(s) in the Event. It is useful when an event leans itself to a compact string representation.
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 @.		

In 5.10.2.3 XML-Syntax replace

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

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

with

```
<!-- 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>
```

5.10.4.1 Overview replace

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

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

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

with

DASH specific events that are of relevance for the DASH client are signalled in the MPD. The URN "urn:mpeg:dash:event:2012" is defined to identify the event scheme defined in Table 22.
For events using this schema, the `emsg`.message_data[]' field will contain the DASHEvent structure defined below:

```
aligned(8) struct DASHEvent
{
   string publish_time; // MPD@publishTime, per sec. 5.10.4.3
   if ( `emsg`.value == 2 )
   {
      string mpd_patch; // MPD patch, per sec. 5.10.4.3
   }
   if ( `emsg`.value == 3 )
   {
      string mpd; // full MPD, per sec. 5.10.4.4
   }
}
```

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

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

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

In 5.10.4.2 MPD validity expiration, 3rd paragraph, replace

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

— the `message_data` field contains the publish time of an MPD, i.e. the value of the **`MPD`**`@publishTime`.

— The media presentation time beyond the event time (indicated time by `presentation_time_delta`) is correctly described only by MPDs with publish time greater than indicated value in the `message_data` field.

— the event duration expresses the remaining duration of Media Presentation from the event time. If the event duration is 0, Media Presentation ends at the event time. If 0xFFFF, the media presentation duration is unknown. In the case in which both `presentation_time_delta` and `event_duration` are zero, then the Media Presentation is ended.

with

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

— the `DASHEvent.publish_time` field contains the publish time of an MPD, i.e. the value of the **`MPD`**`@publishTime`.

— The media presentation time beyond the event time (indicated time by `presentation_time_delta`) is correctly described only by MPDs with publish time greater than indicated value in the `message_data` field.

— the event duration expresses the remaining duration of Media Presentation from the event time. If the event duration is 0, Media Presentation ends at the event time. If 0xFFFF, the media presentation duration is unknown. In the case in which both `presentation_time_delta` and `event_duration` are zero, then the Media Presentation is ended.

In 5.10.4.3 MPD Patch, replace

For DASH events with value 2, an MPD patch shall be included in the message. The payload of this message shall be a valid XML patch compliant to XML Patch Operations framework, as defined in IETF RFC 5261.
The result of the patch application shall be parse-tree identical before any xlink resolution to the MPD that would have been retrieved at event time.
XPath selectors shall at least include a check on **MPD**@publishTime, and the last patch operation should change the value of **MPD**@publishTime.

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

With

For DASH events with value 2, an MPD patch shall be included in theDASHEvent structure, immediately following the publish_time field.. The payload of this message shall be a valid XML patch compliant to XML Patch Operations framework, as defined in IETF RFC 5261.
The result of the patch application shall be parse-tree identical before any XLink resolution to the MPD that would have been retrieved at event time.
XPath selectors shall at least include a check on **MPD**@publishTime, and the last patch operation should change the value of **MPD**@publishTime.

Add 5.10.4.4 MPD Update Event

For DASH events with value 3, a complete MPD shall be included in the in the `DASHEvent` structure, immediately following the `publish_time` field. The content of the `mpd` field shall be the MPD.

In 6.3.4.2 General format type, replace

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

WITH

— The ‘moof’ boxes shall not use external data references, the flag ‘default-base-is-moof’ shall be set, and data-offset shall be used, i.e. ‘base-data-offset-present’ shall not be used. This combination of settings is referred to as movie-fragment relative addressing for media data.

In 7.2.1 Media Presentation timeline, replace

One of the key features in DASH is that encoded versions of different media components share a common timeline. The presentation time of access unit within the media content is mapped to the global common presentation timeline for synchronization of different media components and to enable seamless switching of different coded versions of the same media components.
The presentation times within each Period are relative to the *PeriodStart* time of the Period minus the value of the `@presentationTimeOffset`, T_O, of the containing Representation. This means for an access unit with a presentation time T_P signalled in the media stream, the Media Presentation time relative to the *PeriodStart* is $T_M=T_P - T_O$.
Media Segments should not contain any presentation time T_P that is smaller than the value of the `@presentationTimeOffset`, T_O. However, if this is the case, then presentation of the Media Segment is expected to only take place for presentation times greater than or equal to T_O.
The MPD start times as defined in 5.3.9.5.3 shall provide an approximation of the Media Presentation time T_M within the Period. Specifically, the MPD start time shall be drift-free relative to the presentation time T_P signalled in the media stream, i.e. the accuracy of the offset of the MPD start time relative to the presentation time does not depend on the position of the Segment in the Representation.

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

with

One of the key features in DASH is that encoded versions of different media components share a common timeline. The presentation time of access unit within the media content is mapped to the global common presentation timeline for synchronization of different media components and to enable seamless switching of different coded versions of the same media components.
The presentation times within each Period are relative to the *PeriodStart* time of the Period minus the value of the `@presentationTimeOffset`, T_O, of the containing Representation. This means for an access unit with a presentation time T_P signalled in the media stream, the Media Presentation time relative to the *PeriodStart* is $T_M=T_P - T_O$.
Media Segments should not contain any presentation time T_P that is smaller than the value of the `@presentationTimeOffset`, T_O. However, if this is the case, then presentation of the Media Segment is expected to only take place for presentation times greater than or equal to T_O.
The MPD start times as defined in 5.3.9.5.3 shall provide an approximation of the Media Presentation time T_M within the Period. Specifically, the MPD start time shall be drift-free relative to the presentation time T_P signalled in the media stream, i.e. the accuracy of the offset of the MPD start time relative to the presentation time does not depend on the position of the Segment in the Representation.
At the start of a new Period, the playout procedure of the media content components may need to be adjusted at the end of the preceding Period to match the *PeriodStart* time of the new Period as there may be small overlaps or gaps with a Representation at the end of the preceding Period. Overlaps (respectively gaps) may result from Media Segments with actual presentation duration of the media stream longer (respectively shorter) than indicated by the Period duration. Also in the beginning of a Period if the earliest presentation time T_P of any access unit of a Representation is not equal to T_O then the playout procedures need to be adjusted accordingly.
There may be cases where the Media Presentation author observes issues in generating media, especially for live services. This may be because the input signal to the encoder is not available or the encoder is down. Generally, the Media Presentation author should avoid address these issues by providing redundant architectures or by the use of specific outage or blackout signals. Also, if the format in use permits, empty Segments or zero duration Segments may be used.
However, if such remedies are not available the Media Presentation author may signal gaps in the timeline. If the Segment Timeline is in use as defined in 5.3.9.6, the gaps in the timeline may be explicitly signaled. Alternatively, gaps in the segment timeline may be signaled by leaving gaps.
If no such means are available, early termination Periods may be used as defined in section 5.3.2. In this case, the gaps at the end of the Period may be longer than indicated above. A client is expected to overcome such outages and continue the Media Presentation with the availability of a new Period.

Add 7.3.5 Segment Timeline with Time Templating without Segment Index

If the Segment Timeline is in use and the `$Time$` templating is applied and no Segment Index (‘`sidx`’) box is present in the Media Segment, then

- a single track shall be present in the Media Segment
- a single movie fragment header shall be present in the Media Segment
- the `baseMediaDecodeTime` in the '`tfdt`' of the first movie fragment shall be the earliest presentation time of the Segment and may be used for generating the URL for this segment— the sum of all `sample_duration` of track run boxes (‘`trun`’) of the only track fragment box shall be the presentation duration of the Segment and may therefore be used to derive the address of the next Media Segment from the actual Media Segment without requiring an updated MPD.

In 7.4.1 Introduction, add the following at the end

If the Segment Timeline is in use and the `$Time$` templating is applied, then the Segment Index should be present. Otherwise, the DASH access may have to update the MPD for each Segment request, or it may have to predict the earliest presentation time and the duration of the Segment by parsing the full Segment in order to obtain the URL of the next Segment.

In A.3.3 Start Time and Duration, in 5th and 6th paragraph, replace

If the Representation contains or inherits a **`SegmentTemplate`** element with *$Number$* then the URL of the Media Segment *i*, MediaSegment.URL[*i*], is obtained by replacing the $*Number*$ identifier by *i* + `@startNumber` in the **`SegmentTemplate`**`@media` string.
If the Representation contains or inherits a **`SegmentTemplate`** element with *$Time$* then the URL of the Media Segment *i*, MediaSegment.URL[*i*], is obtained by replacing the $*Time*$ identifier by MediaSegment[*i*].startTime in the **`SegmentTemplate`**`@media` string.

with

If the Representation contains or inherits a **`SegmentTemplate`** element with *$Number$* then the URL of the Media Segment *i*, MediaSegment.URL[*i*], is obtained by replacing the $*Number*$ identifier by *i - 1* + `@startNumber` in the **`SegmentTemplate`**`@media` string.
If the Representation contains or inherits a **`SegmentTemplate`** element with *$Time$* then the URL of the Media Segment *i*, MediaSegment.URL[*i*], is obtained by replacing the $*Time*$ identifier by MediaSegment[*i*].startTime in the **`SegmentTemplate`**`@media` string, as described in 5.3.9.5.3.

In Annex B, replace

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

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

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

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

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

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

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

  <!-- Event Stream -->
  <xs:complexType name="EventStreamType">
```

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


  <!-- Event  -->
  <xs:complexType name="EventType">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="presentationTime" type="xs:unsignedLong" default="0"/>
        <xs:attribute name="duration" type="xs:unsignedLong"/>
        <xs:attribute name="id" type="xs:unsignedInt"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

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

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

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

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

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

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

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


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

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

```
    <xs:attribute name="width" type="xs:unsignedInt"/>
    <xs:attribute name="height" type="xs:unsignedInt"/>
    <xs:attribute name="sar" type="RatioType"/>
    <xs:attribute name="frameRate" type="FrameRateType"/>
    <xs:attribute name="audioSamplingRate" type="xs:string"/>
    <xs:attribute name="mimeType" type="xs:string"/>
    <xs:attribute name="segmentProfiles" type="xs:string"/>
    <xs:attribute name="codecs" type="xs:string"/>
    <xs:attribute name="maximumSAPPeriod" type="xs:double"/>
    <xs:attribute name="startWithSAP" type="SAPType"/>
    <xs:attribute name="maxPlayoutRate" type="xs:double"/>
    <xs:attribute name="codingDependency" type="xs:boolean"/>
    <xs:attribute name="scanType" type="VideoScanType"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

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

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

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

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

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

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