				smooth playout provided that each Representation is delivered at or above the value of its *bandwidth* attribute.
timeShiftBufferDepth	A		O	Indicates the duration of the time shifting buffer that is available for a live presentation. When not present, the value is unknown. If present for on-demand services, this attribute shall be ignored by the client.
baseURL	A		O	Base URL on MPD level
ProgramInformation	E	0,1	O	Provides descriptive information about the program
moreInformationURL	A		O	This attribute contains an absolute URL which provides more information about the Media Presentation
Title	E	0,1	O	May be used to provide a title for the Media Presentation
Source	E	0,1	O	May be used to provide information about the original source (for example content provider) of the Media Presentation.
Copyright	E	0,1	O	May be used to provide a copyright statement for the Media Presentation.
Period	E	1…N	M	Provides the information of a Period
start	A		M	Provides the accurate start time of the Period relative to the value of the attribute *availabilityStart* time of the Media Presentation.
segmentAlignmentFlag	A		O Default: false	When True, indicates that all start and end times of media components of any particular media type are temporally aligned in all Segments across all Representations in this Period.
bitstreamSwitchingFlag	A		O Default: false	When True, indicates that the result of the splicing on a bitstream level of any two time-sequential Media Segments within a Period from any two different Representations containing the same media types complies to the Media Segment format.
SegmentInfoDefault	E	0,1	O	Provides default Segment information about Segment durations and, optionally, URL construction.
duration	A		O	Default duration of Media Segments
baseURL	A		O	Base URL on Period level
sourceUrlTemplatePeriod	A		O	The source string providing the URL template on period level.
Representation	E	1..N	M	This element contains a description of a Representation.
bandwidth	A		M	The minimum bandwidth of a hypothetical constant bitrate channel in bits per second (bps) over which the representation can be delivered such that a client, after buffering for exactly *minBufferTime* can be assured of having enough data for continuous playout.
width	A		O	Specifies the horizontal resolution of the video media type in an alternative Representation, counted in pixels.
height	A		O	Specifies the vertical resolution of the video media type in an alternative Representation, counted in pixels.
lang	A		O	Declares the language code(s) for this Representation according to RFC 5646 [106]. Note, multiple language codes may be declared when e.g. the audio and the sub-title are of different languages.
mimeType	A		M	Gives the MIME type of the Initialisation Segment, if present; if the Initialisation Segment is not present it provides the MIME type of the first Media Segment. Where applicable, this MIME type includes the codec parameters for all media types. The codec parameters also include the profile and level information where applicable.

				For 3GP files, the MIME type is provided according to RFC 4281 [107].
group	A		OD Default: 0	Specifies the group to which this Representation is assigned.
startWithRAP	A		OD Default: False	When True, indicates that all Segments in the Representation start with a random access point
qualityRanking	A		O	Provides a quality ranking of the Representation relative to other Representations in the Period. Lower values represent higher quality content. If not present then the ranking is undefined.
ContentProtection	E	0, 1	O	This element provides information about the use of content protection for the segments of this representation. When not present the content is not encrypted or DRM protected.
SchemeInformation	E	0,1	O	This element gives the information about the used content protection scheme. The element can be extended to provide more scheme specific information.
schemeIdUri	A		O	Provides an absolute URL to identify the scheme. The definition of this element is specific to the scheme employed for content protection.
TrickMode	E	0, 1	O	Provides the information for trick mode. It also indicates that the Representation may be used as a trick mode Representation.
alternatePlayoutRate	A		O	Specifies the maximum playout rate as a multiple of the regular playout rate, which this Representation supports with the same decoder profile and level requirements as the normal playout rate.
SegmentInfo	E	1		Provides Segment access information.
duration	A		CM Must be present in case duration is not present on period level and the Representation contains more than one Media Segment.	If present, gives the constant approximate segment duration. The attribute must be present in case duration is not present on Period level and the Representation contains more than one Media Segment. If the Representation contains more only one Media Segment, then this attribute may not be present. All Segments within this SegmentInfo element have the same duration unless it is the last Segment within the Period, which could be significantly shorter.
baseURL	A		O	Base URL on Representation level
InitialisationSegmentURL	E	0, 1	O	This element references the Initialisation Segment. If not present each Media Segment is self-contained.
sourceURL	A		M	The source string providing the URL
range	A		O	The byte range restricting the above URL. If not present, the resources referenced in the sourceURL are unrestricted. The format of the string shall comply with the format as specified in section 12.2.4.1.
UrlTemplate	E	0, 1	CM Must be present if the Url element is not present.	The presence of this element specifies that a template construction process for Media Segments is applied. The element includes attributes to generate a Segment list for the Representation associated with this element.
sourceURL	A		O	The source string providing the template. This attribute and the *id* attribute are mutually exclusive.
id	A		CM Must be present if the sourceUrlTemplatePeriod attribute is present	An attribute containing a unique ID for this specific Representation within the Period. This attribute and the *sourceURL* attribute are mutually exclusive.

	startIndex	A		OD default: 1	The index of the first accessible Media Segment in this Representation. In case of on-demand services or in case the first Media Segment of the Representation is accessible, then this value shall not be present or shall be set to 1.
	endIndex	A		O	The index of the last accessible Media Segment in this Representation. If not present the *endIndex* is unknown.
Url		E	0 …N	CM Must be present if the UrlTemplate element is not present.	Provides a set of explicit URL(s) for Segments. Note: The URL element may contain a byte range.
	sourceURL	A		M	The source string providing the URL
	range	A		O	The byte range restricting the above URL. If not present, the resources referenced in the sourceURL are unrestricted. The format of the string shall comply with the format as specified in section 12.2.4.1

12.2.5.3 Media Presentation Description Schema

The XML schema for the MPD in section 12.2.5.2 is as follows:

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

<!-- MPD Type -->
<xs:complexType name="MPDtype">
   <xs:sequence>
      <xs:element minOccurs="0" name="ProgramInformation" type="ProgramInformationType"/>
      <xs:element maxOccurs="unbounded" name="Period" type="PeriodType"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute default="OnDemand" name="type" type="PresentationType"/>
   <xs:attribute name="availabilityStartTime" type="xs:dateTime"/>
   <xs:attribute name="availabilityEndTime" type="xs:dateTime"/>
   <xs:attribute name="mediaPresentationDuration" type="xs:duration"/>
   <xs:attribute name="minimumUpdatePeriodMPD" type="xs:duration"/>
   <xs:attribute name="minBufferTime" type="xs:duration" use="required"/>
   <xs:attribute name="timeShiftBufferDepth" type="xs:duration"/>
   <xs:attribute name="baseUrl" type="xs:anyURI"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

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

<!-- Period of a presentation -->
<xs:complexType name="PeriodType">
   <xs:sequence>
      <xs:element minOccurs="0" name="SegmentInfoDefault" type="SegmentInfoDefaultType"/>
      <xs:element maxOccurs="unbounded" name="Representation" type="RepresentationType"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
```

```
   <xs:attribute name="start" type="xs:duration"/>
   <xs:attribute default="false" name="segmentAlignmentFlag" type="xs:boolean"/>
   <xs:attribute default="false" name="bitStreamSwitchingFlag" type="xs:boolean"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

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

<!-- Default Segment access information -->
<xs:complexType name="SegmentInfoDefaultType">
   <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="baseURL" type="xs:anyURI"/>
   <xs:attribute name="duration" type="xs:duration"/>
   <xs:attribute name="sourceUrlTemplatePeriod" type="xs:string"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- A Representation of the presentation content for a specific Period -->
<xs:complexType name="RepresentationType">
   <xs:sequence>
      <xs:element name="SegmentInfo" type="SegmentInfoType"/>
      <xs:element minOccurs="0" name="ContentProtection" type="ContentProtectionType"/>
      <xs:element minOccurs="0" name="TrickMode" type="TrickModeType"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="bandwidth" type="xs:unsignedInt" use="required"/>
      <xs:attribute default="0" name="group" type="xs:unsignedInt"/>
   <xs:attribute name="width" type="xs:unsignedInt"/>
   <xs:attribute name="height" type="xs:unsignedInt"/>
   <xs:attribute name="lang" type="xs:string"/>
   <xs:attribute name="mimeType" type="xs:string" use="required"/>
   <xs:attribute default="false" name="startWithRAP" type="xs:boolean"/>
   <xs:attribute name="qualityRanking" type="xs:unsignedInt"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- Segment access information -->
<xs:complexType name="SegmentInfoType">
   <xs:sequence>
      <xs:element minOccurs="0" name="InitialisationSegmentURL" type="UrlType"/>
      <xs:choice>
         <xs:element minOccurs="0" name="UrlTemplate" type="UrlTemplateType"/>
         <xs:sequence>
            <xs:element maxOccurs="unbounded" name="Url" type="UrlType"/>
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
               maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:any namespace="##other" processContents="lax" minOccurs="0"
            maxOccurs="unbounded"/>
      </xs:choice>
   </xs:sequence>
   <xs:attribute name="baseURL" type="xs:anyURI"/>
   <xs:attribute name="duration" type="xs:duration"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

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

```
<!-- A URL template  -->
<xs:complexType name="UrlTemplateType">
   <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="sourceURL" type="xs:anyURI"/>
   <xs:attribute name="id" type="xs:string"/>
   <xs:attribute default="1" name="startIndex" type="xs:unsignedInt"/>
   <xs:attribute name="endIndex" type="xs:unsignedInt"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- Gives information about the content protection -->
<xs:complexType name="ContentProtectionType">
   <xs:sequence>
      <xs:element minOccurs="0" name="SchemeInformation" type="xs:string"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="schemeIdUri" type="xs:anyURI"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- Gives information about trick mode -->
<xs:complexType name="TrickModeType">
   <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="alternatePlayoutRate" type="xs:string"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

</xs:schema>
```

An example for a valid MPD is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
   type="Live"
   baseUrl="http://www.example.com"
   minimumUpdatePeriodMPD="PT20S"
   minBufferTime="PT10S"
   mediaPresentationDuration="PT2H"
   availabilityStartTime="2010-04-01T09:30:47Z"
   availabilityEndTime="2010-04-07T09:30:47Z"
   timeShiftBufferDepth="PT30M"
   xsi:schemaLocation="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009 3GPP-MPD-r1.xsd"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
   <ProgramInformation moreInformationURL="http://www.example.com">
      <Title>Example</Title>
   </ProgramInformation>
   <Period start="PT0S">
      <Representation
         mimeType="video/3gpp; codecs=s263, samr"
         bandwidth="256000">
         <SegmentInfo duration="PT10S" baseURL="rep1/">
            <InitialisationSegmentURL sourceURL="seg-init.3gp"/>
            <Url sourceURL="seg-1.3gp"/>
            <Url sourceURL="seg-2.3gp"/>
            <Url sourceURL="seg-3.3gp"/>
         </SegmentInfo>
      </Representation>
      <Representation
         mimeType="video/3gpp; codecs=mp4v.20.9, mp4a.E1"
         bandwidth="128000">
         <SegmentInfo duration="PT10S" baseURL="rep2/">
            <InitialisationSegmentURL sourceURL="seg-init.3gp"/>
            <Url sourceURL="seg-1.3gp"/>
            <Url sourceURL="seg-2.3gp"/>
            <Url sourceURL="seg-3.3gp"/>
         </SegmentInfo>
      </Representation>
   </Period>
   <Period start="PT30S">
      <SegmentInfoDefault
```

```
        duration="PT10S"
        sourceUrlTemplatePeriod="http://example.com/$RepresentationId$/$Index$.3gp"/>
      <Representation
        mimeType="video/3gpp; codecs=mp4v.20.9, mp4a.E1"
        bandwidth="256000">
        <SegmentInfo>
          <UrlTemplate id="1"/>
        </SegmentInfo>
      </Representation>
      <Representation
        mimeType="video/3gpp; codecs=mp4v.20.9, mp4a.E1"
        bandwidth="128000">
        <SegmentInfo>
          <UrlTemplate id="2"/>
        </SegmentInfo>
      </Representation>
    </Period>
</MPD>
```

12.2.5.4 Media Presentation Description Updates

The MPD may be changed during the Media Presentation.

If the attribute *minimumUpdatePeriodMPD* is provided then the client derives the next time to check whether the MPD has been updated on the server, referred to as *CheckTime*, as the sum of the time when the previous update of the MPD was requested and the *minimumUpdatePeriodMPD*.

If the *minimumUpdatePeriodMPD* is not provided, external means may be used to deduce the *CheckTime* of the MPD, if applicable.

If the MPD is changed, then the changes to the MPD must be such that the updated MPD is compatible with the previous MPD in the following sense: An example client using the segment list generation process as provided in section 12.6.3 would generate an identically functional Segment List from the updated MPD for any time up to the *CheckTime* of the previous MPD as it would have done from the previous MPD. The requirement ensures that

1. clients may immediately begin using the new MPD without synchronisation with the old MPD, since it is compatible with the old MPD before the update time; and

2. the update time needs not be synchronised with the time at which the actual change to the MPD takes place: i.e. changes to the MPD may be advertised in advance.

12.3 Protocols

Adaptive HTTP-Streaming Clients shall comply with clients as specified in RFC2616 [17].

HTTP-Streaming Servers shall comply with servers as specified in RFC2616 [17].

HTTP-Streaming Clients is expected to use the HTTP GET method or the partial GET method, as specified in RFC2616 [17], section 9.3, to retrieve Initialisation Segments or parts thereof and Media Segments or parts thereof.

12.4 Usage of 3GPP File Format

12.4.1 Instantiation of 3GPP Adaptive HTTP Streaming

The Media Presentation framework as introduced in section 12.2 is instantiated in this section using the 3GP File Format as specified in [50]. This instantiation is referred to as '3GPP Adaptive HTTP Streaming'. A 3GPP Adaptive HTTP Streaming service is described by an MPD as specified in section 12.2.5. The MIME type of the MPD shall be 'video/vnd.3gpp.mpd'. The *mimeType* attribute of each Representation shall be provided according to RFC 4281 [107] and Annex A of TS 26.244 [50]. Segment Types and Formats according to 12.4.2 shall be used.

12.4.2 Segment Types and Formats

12.4.2.1 Segment Types

3GPP Adaptive HTTP Streaming defines a Segment format that is used in the delivery of media data over HTTP. A Segment shall contain one or more boxes in accordance with the boxed structure of the ISO-base media file format [104].

Three different Segment types are defined for 3GPP adaptive HTTP streaming.

1. Initialisation Segment with a MIME type as defined in Annex A of TS 26.244 [50] .
2. Media Segment with a MIME type 'video/vnd.3gpp.segment' if accessed through a URL without byte ranges or as defined in Annex A of TS 26.244 [50] if accessed through a URL with byte ranges.
3. Self-Initialising Media Segment with a MIME type as defined in Annex A of TS 26.244 [50] .

NOTE: the MIME type for Media Segments is defined in Annex A.1.4 of TS 26.244 [50].

For 3GPP adaptive HTTP streaming the following applies:

- In all cases for which a Representation contains more than one Media Segment, the following applies:
  - The Initialisation Segment as defined in section 12.4.2.2 shall be present. The Initialisation Segment shall be available to the HTTP Streaming Client before any Media Segment is processed within the Representation.
  - Media Segments shall not be self-initialising. The Media Segment format is defined in section 12.4.2.3.
- In case a Representation contains only a single Media Segment, then either one of the following two options is used:
  1. An Initialisation Segment as defined in section 12.4.2.2 and one Media Segment as defined in section 12.4.2.3.
  2. One Self-Initialising Media Segment as defined in section 12.4.2.4.

12.4.2.2 Initialisation Segment Format

The Initialisation Segment is conformant with the 3GPP file format, adaptive streaming profile and shall be branded as '3gh9'.

The Initialisation Segment consists of the 'ftyp' box, the 'moov' box, and optionally the 'pdin' box. The 'moov' box contains no samples (i.e. the entry_count in the 'stts', 'stsc', and 'stco' boxes shall be set to 0) and is then very small in size. This reduces the start-up time significantly as the Initialisation Segment needs to be downloaded before any Media Segment can be processed.

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

The Initialisation Segment provides the client with the metadata that describes the media content. The client uses the information in the 'moov' box to identify the available media components and their characteristics.

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

12.4.2.3 Media Segment Format

The following constraints shall apply to a Media Segment conforming to Media Segment Format for 3GPP adaptive HTTP streaming:

- Each Media Segment may contain an "styp" box.

- Each Media Segment contains one or more whole self-contained movie fragments. A whole, self-contained movie fragment is a movie fragment ("moof") box and a media data ("mdat") box that contains all the media samples referenced by the track runs in the movie fragment box.
- Each "moof" box shall contain at least one track fragment.
- The "moof" boxes shall use movie-fragment relative addressing. Absolute byte-offsets shall not be used. In a movie fragment, the durations by which each track extends should be as close to equal as practical. In particular, as movie fragments are accumulated, the track durations should remain close to each other and there should be no 'drift'.
- Each "traf" box may contain a "tfad" box.
- Each Media Segment may contain one or more "sidx" boxes. If present, the first "sidx" box shall be placed before any "moof" box and the subsegment documented by the first Segment Index box shall be the entire segment.

12.4.2.4 Self-Initialising Media Segment Format

The following constraints shall apply to a Media Segment conforming to Self-Initialising Media Segment Format for 3GPP adaptive HTTP streaming:

- The Self-Initialising Media Segment shall comply to the 3GP Adaptive-Streaming profile as specified in TS 26.244 [50].
- Movie fragment ("moof") boxes may be present
- Each "moof" box, if present, shall contain at least one track fragment.
- The "moof" boxes, if present, shall use movie-fragment relative addressing. Absolute byte-offsets shall not be used. In a movie fragment, the durations by which each track extends should be as close to equal as practical. In particular, as movie fragments are accumulated, the track durations should remain close to each other and there should be no 'drift'.
- Each "traf" box, if present, may contain a "tfad" box.
- A Self-Initialising Media Segment may contain one or more "sidx" boxes. If present, the first "sidx" box shall be placed before any "moof" box and the subsegment documented by the first "sidx" shall be the entire Media Segment.

12.4.3 Usage on Server and Client

3GPP Adaptive HTTP-Streaming uses 3GP files according to the 3GP Adaptive-Streaming profile as specified in TS 26.244 [50]. Content may be prepared as 3GP files according to the 3GP Adaptive-Streaming profile. Initialisation Segments and Media Segments may be generated by segmenting such 3GP files. Segment Index "sidx" boxes may be pre-contained in 3GP files or may be generated during the segmentation process. Clients may store a concatenation of a received Initialisation Segment and a sequence of Media Segments to create a compliant 3GP file according to the Adaptive Streaming profile without accessing any media samples.

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

12.5 Media Codecs

For HTTP-Streaming clients supporting a particular continuous media type, the corresponding media decoders are specified in section 7.2 for speech, 7.3 for audio, 7.4 for video and 7.9 for timed text.

12.6 Client Behaviour

12.6.1 Introduction

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

12.6.2 Overview

A 3GPP Adaptive HTTP Streaming client is guided by the information provided in the MPD. It is assumed that the client has access to the MPD. An example client behaviour is introduced. For providing a continuous streaming service to the user:

1. The client parses the MPD and creates a list of accessible Segments for each Representation for the actual client-local time *NOW* taking into account the procedures specified in section 12.6.3.

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

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

4. The client buffers media of at least *minBufferTime* duration before starting the presentation. Once the presentation has started, the client continues consuming the media content by continuously requesting Media Segments or parts of Media Segments taking into account the MPD update and construction procedures in clause 12.6.3. The client may switch Representations taking into account updated MPD information and/or updated information from its environment, e.g. change of available bandwidth. With any request for a Media Segment containing a random access point, the client may switch to a different Representation.

In the following a brief overview on Segment list generation, seeking, support for trick modes and switching Representations are provided.

12.6.3 Segment List Generation

12.6.3.1 General

Assume that the HTTP-streaming client has access to an MPD. This section describes how a client may generate a Segment list as shown in Table 12.3 from an MPD at a specific client-local time *NOW*. In this description, the term *NOW* is used to refer to 'the current value of the clock at the reference client when performing the construction of an MPD Instance from an MPD'. A client that is not synchronised with a HTTP Streaming server, which is in turn synchronised to UTC, may experience issues in accessing segments due to availability. HTTP Streaming clients should synchronize their clocks to a globally accurate time standard.

Table 12.3 Segment List

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

Each *SegmentInfo* element is used to generate a list of accessible Segments for each Representation.

The following rules apply for *SegmentInfoDefault* elements or *SegmentInfo* elements in a MPD:

- The client uses URI reference resolution as discussed in section 12.2.4.2.1. If the MPD is known to be supplied using a URL and over a suitable protocol, that URL establishes a base URL for the segments URLs within the MPD. There may be a *baseURL* attribute on MPD level or in the *SegmentInfoDefault* element on Period level or the *SegmentInfo* element. If the *baseURL* attribute supplied at any level is absolute, it gives the base URL for the levels below it. Otherwise the base URL for levels below it is formed from the base URL of the higher level composed with the value of the *baseURL* attribute. Normal URL composition may be used, using relative URLs, which are composed against a base URL. The composition of a relative URL with an effective base URL is done using normal URL Reference Resolution (see RFC 3986 [60], section 5.2).
- If the *SegmentInfo* element contains a *URLtemplate* element, then the procedures in section 12.6.3.2 are used to generate a list of Media Segments.
- If the *SegmentInfo* element contains one or more *Url* elements providing a set of explicit URL(s) for Media Segments, then the procedures in section 12.6.3.3 are used to generate a list of Media Segments.
- If the *type* attribute of the MPD is Live, then the restrictions on Media Segment Lists as provided in section 12.6.3.4.4 need to be taken into account.

The client should only request Segments that are included in the segment list at time instant *NOW*.

12.6.3.2 Template-based Generation of Media Segment List

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

The Segment information for a Representation is obtained by combining the *SegmentInfo* element with the *SegmentInfoDefault* element on Period level. The *duration* attribute of the *SegmentInfo* element overrides the same attribute of the *SegmentInfoDefault* element on Period level.

Assume that the Period duration is defined as *PeriodDuration*. For any Period in the MPD except for the last one, the *PeriodDuration* is obtained as the time difference between the *start* attribute of the next Period and the *start* attribute of the current Period. For the last Period in the MPD, the *PeriodDuration* is obtained as the time difference between the *CheckTime* as defined in section 12.2.5.4 or the end time of the Media Presentation and the UTC start time of the current Period.

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

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

A valid list of Media Segments with Segment indices *Index*[i], MediaSegment.StartTime[i] and MediaSegment.URL[i], i=1,2,3, …, is obtained as follows using the *duration* attribute for this Representation:

1. Set i=1.
2. The start index of the first Media Segment is set to *startIndex*, i.e. *Index*[1]=*startIndex*. The start time of the first Media Segment is obtained as (*startIndex*-1)**duration*, i.e. MediaSegment.StartTime[1] = (*startIndex*-1)**duration*.
3. The URL of the Media Segment *i*, MediaSegment.URL[i], is obtained by replacing the $*Index$* identifier by *Index*[i] in the *sourceURL* string of the valid *UrlTemplate*. Furthermore, any relative URLs are resolved as specified in section 12.2.4.2.1.

4. If ((*PeriodDuration* - MediaSegment.StartTime[*i*] - *duration*) >= *duration)* and ((*Index*[*i*]+1) <= *endIndex*))
   - then
     - A new Media Segment is added to the list, i.e. *i* = *i* + 1;
     - MediaSegment.StartTime[*i*] = MediaSegment.StartTime[*i*-1] + *duration*.
     - *Index*[i] = *Index*[i-1] + *1*
     - Proceed with step 3.
   - else
     - The restrictions as specified in section 12.6.3.4 are applied for the creation of the accessible list of Media Segments.

12.6.3.3 Playlist-based Generation of Media Segment List

If the *SegmentInfo* element contains one or more *Url* elements, then the procedures specified in this section apply to generate a valid list of accessible Media Segment URLs and start times described in each *SegmentInfo* element taking into account the procedures to integrate information from *SegmentInfoDefault* elements.

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

A valid list of Media Segments with segment indices *i*=1,2,3, …, MediaSegment.StartTime[*i*] and MediaSegment.URL[*i*] is obtained as follows:

1. Set *i*=1.
2. The URL of the Media Segment *i*, MediaSegment.URL[*i*], is obtained as the *sourceURL* attribute of the *i*-th *Url* element in the *SegmentInfo* element taking into account URI reference resolution, restricted to the byte range specified in the *range* attribute of the same *Url* element, if present.
3. If the *duration* attribute is provided, then the MediaSegment.StartTime[*i*] of Media Segment *i* is obtained as (*i*-1)**duration*. If the *duration* attribute is not provided, then the MediaSegment.StartTime[*1*] of the only provided Segment is set to 0.
4. If this is not the last Url element, a new Media Segment is added to the list, i.e. *i* = *i* + 1, and proceed with step 2; Otherwise proceed with step 5.
5. The restrictions as specified in section 12.6.3.4 are applied for the creation of the accessible list of Media Segments.

12.6.3.4. Media Segment List Restrictions

The Media Segment List is restricted to t a list of accessible Media Segments, which may be a subset of the Media Segments of the complete Media Presentation. The construction is governed by the current value of the clock at the client *NOW*.

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

In addition, for live services, the Media Segment list is further restricted by the *CheckTime* as defined in section 12.2.5.4 together with the MPD attributes *timeShiftBufferDepth* such that only Media Segments for which the sum of the start time of the Media Segment and the Period start time falls in the interval [*NOW-timeShiftBufferDepth-duration,* min(*CheckTime, NOW*))] are included.

12.6.4 Seeking

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

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

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

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

Note that timing information in the MPD may be approximate due to issues related to placement of Random Access Points, alignment of media tracks and media timing drift. As a result, the Segment identified by the procedure above may begin at a time slightly after *tp* and the media data for presentation time *tp* may be in the previous Media Segment. In case of seeking, either the seek time may be updated to equal the first sample time of the retrieved file, or the preceeding file may be retrieved instead. However, note that during continuous playout, including cases where there is a switch between alternative versions, the media data for the time between *tp* and the start of the retrieved Segment is always available.

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

Also note that not necessarily all information of the Media Segment needs to be downloaded to access the presentation time *tp*. The client may for example initially request the "sidx" box from the beginning of the Media Segment using byte range requests. By use of the "sidx", segment timing can be mapped to byte ranges of the Segment. By continuously using partial HTTP requests, only the relevant parts of the Media Segment may be accessed for improved user experience and low start-up delays.

12.6.5 Support for Trick Modes

The client may pause or stop a Media Presentation. In this case client simply stops requesting Media Segments or parts thereof. To resume, the client sends requests to Media Segments, starting with the next fragment after the last requested fragment.

If the MPD for a specific Representation contains the *TrickMode* element, then this Representation is explicitly enabled for the use with trick modes. The client may play the Representation with any speed up to the regular speed times the specified *alternatePlayoutRate* attribute with the same decoder profile and level requirements as the normal playout rate.

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

12.6.6 Switching Representations

Based on updated information during an ongoing Media Presentation, a client may decide to switch Representations. Switching to a 'new' Representation is equivalent to tuning in or seeking to the new Representation from the time point where the "old" Representation has been presented. Once switching is desired, the client should seek to a RAP in the 'new' Representation at a desired presentation time *tp* later than and close to the current presentation time. Presenting the 'old' Representation up to the RAP in the 'new' Representation enables seamless switching.

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

12.6.7 Reaction to Error Codes

The HTTP Streaming client provides a streaming service to the user by issuing HTTP requests for Segments at appropriate times. The HTTP Streaming client may also update the MPD by using HTTP requests. In regular operation mode, the server typically responds to such requests with status code 200 OK (for regular GET) or status code 206 Partial Content (for partial GET) and the entity corresponding to the requested resource. Other Successful 2xx or Redirection 3xx status codes may be returned.

HTTP requests may result in a Client Error 4xx or Server Error 5xx status code. Some guidelines are provided in this section as to how an HTTP client may react to such error codes.

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

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

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

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

Upon receiving server errors (i.e. messages with 5xx error code), the client should check for an update of the MPD. The client may also check for alternative representations that are hosted on a different server.

12.7 Security

12.7.1 Content Protection

Clients that support content protection may support OMA DRM 2.0 [75] or OMA DRM 2.1 [109] as suitable for content protection. Other DRM or encryption schemes may be supported. The ContentProtection element in the MPD should be used to convey content protection information.

When OMA DRM is supported for content protection, the non-streamable Packetized DRM Content Format (PDCF) shall be used.

An OMA-DRM encrypted Representation shall include the brands '3gh9' and 'opf2'.

OMA-DRM [74] defines the procedures for acquiring the Rights Object from the Rights Issuer to decrypt PDCF protected content.

12.7.2 Transport Security

Transport security in adaptive HTTP streaming is achieved using the HTTPS (Hypertext Transfer Protocol Secure) specified in RFC2818 [105]. HTTPS may be used to authenticate the server and to ensure secure transport of the content from server to client.

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

Annex A (informative): Protocols

A.1 SDP

This clause gives some background information on SDP for PSS clients.

Table A.1 provides an overview of the different SDP fields that can be identified in a SDP file. The order of SDP fields is mandated as specified in RFC 4566 [6].

Table A.1: Overview of fields in SDP for PSS clients

Type	Description		Requirement according to [6]	Requirement according to the present document
Session Description				
V	Protocol version		R	R
O	Owner/creator and session identifier		R	R
S	Session Name		R	R
I	Session information		O	O
U	URI of description		O	O
E	Email address		O	O
P	Phone number		O	O
C	Connection Information		R	R
B	Bandwidth information	AS	O	O
		RS	ND	O
		RR	ND	O
		TIAS	ND	O
One or more Time Descriptions (See below)				
Z	Time zone adjustments		O	O
K	Encryption key		O	O
A	Session attributes	control	O	R
		range	O	R
		alt-group	ND	O
		3GPP-QoE-Metrics	ND	O
		3GPP-Asset-Information	ND	O
		3GPP-Integrity-Key	ND	O
		3GPP-SDP-Auth	ND	O
		maxprate	ND	O
One or more Media Descriptions (See below)				
Time Description				
T	Time the session is active		R	R
R	Repeat times		O	O
Media Description				
M	Media name and transport address		R	R
I	Media title		O	O
C	Connection information		R	R
B	Bandwidth information	AS	O	R
		RS	ND	R
		RR	ND	R
		TIAS	ND	R
K	Encryption Key		O	O
A	Attribute Lines	control	O	R
		range	O	R
		fmtp	O	R
		rtpmap	O	R
		X-predecbufsize	ND	O
		X-initpredecbufperiod	ND	O
		X-initpostdecbufperiod	ND	O
		X-decbyterate	ND	O
		framesize	ND	R (see note 5)
		alt	ND	O
		alt-default-id	ND	O
		3GPP-Adaptation-Support	ND	O
		3GPP-QoE-Metrics	ND	O
		3GPP-Asset-Information	ND	O
		3GPP-SRTP-Config	ND	O
		rtcp-fb	O	O
		maxprate	ND	R

Note 1: R = Required, O = Optional, ND = Not Defined

Note 2: The "c" type is only required on the session level if not present on the media level.

Note 3: The "c" type is only required on the media level if not present on the session level.

Note 4: According to RFC 4566, either an 'e' or 'p' field must be present in the SDP description. On the other hand, both fields will be made optional in the future release of SDP. So, for the sake of robustness and maximum interoperability, either an 'e' or 'p' field shall be present during the server's SDP file creation, but the client should also be ready to receive SDP content containing neither 'e' nor 'p' fields.

Note 5: The "framesize" attribute is only required for H.263 streams.

Note 6: The 'range' attribute is required on either session or media level: it is a session-level attribute unless the presentation contains media streams of different durations. If a client receives 'range' on both levels, however, media level shall override session level.

The example below shows an SDP file that could be sent to a PSS client to initiate unicast streaming of a H.263 video sequence.

EXAMPLE 1:

```
v=0
o=ghost 2890844526 2890842807 IN IP4 192.168.10.10
s=3GPP Unicast SDP Example
i=Example of Unicast SDP file
u=http://www.infoserver.com/ae600
e=ghost@mailserver.com
c=IN IP4 0.0.0.0
t=0 0
a=range:npt=0-45.678
m=video 1024 RTP/AVP 96
b=AS:56
b=TIAS:52500
a=maxprate:11
a=rtpmap:96 H263-2000/90000
a=fmtp:96 profile=3;level=10
a=control:rtsp://mediaserver.com/movie.3gp/trackID=1
a=framesize:96 176-144
```

The following examples show some usage of the "alt" and the "alt-default-id" attributes (only the affected part of the SDP is shown):

EXAMPLE 2:

```
m=audio 0 RTP/AVP 97
b=AS:12
b=TIAS:8500
a=maxprate:10
a=rtpmap:97 AMR/8000
a=control:trackID=1
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
a=alt-default-id:1
a=alt:2:b=AS:16
a=alt:2:b=TIAS:12680
a=alt:2:a=control:trackID=2
```

The equivalent SDP for alternative 1 (default) is:

EXAMPLE 3:

```
m=audio 0 RTP/AVP 97
b=AS:12
b=TIAS:8500
a=maxprate:10
a=rtpmap:97 AMR/8000
```

```
a=control:trackID=1
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
```

Alternative 2 is based on the default alternative but replaces two lines, "b=AS" and "a=control". Hence, the equivalent SDP for alternative 2 is:

EXAMPLE 4:
```
m=audio 0 RTP/AVP 97
b=AS:16
b=TIAS:12680
a=maxprate:10
a=rtpmap:97 AMR/8000
a=control:trackID=2
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
```

Below is an example on the usage of the "alt-group" attribute with the subtype "BW":

EXAMPLE 5:
```
a=alt-group:BW:AS:32=1,4;56=2,4;64=3,5
```

The above line gives three groupings based on application-specific bitrate values. The first grouping will result in 32 kbps using media alternatives 1 and 4. The second grouping has a total bitrate of 56 kbps using media alternatives 2 and 4. The last grouping needs 64 kbps when combing media alternatives 3 and 5.

Here follows an example on the usage of the "alt-group" attribute with the subtype "LANG":

EXAMPLE 6:
```
a=alt-group:LANG:RFC3066:en-US=1,2,4,5;se=3,4,5
```

The above line claims that the media alternatives 1, 2, 4, and 5 support US English and that the media alternatives 3, 4, and 5 support Swedish.

A more complex example where a combination of "alt", "alt-default-id" and "alt-group" are used is seen below. The example allows a client to select a bandwidth that is suitable for the current context in an RTSP SETUP message. The client sends an RTSP DESCRIBE to the server and the server responds with the following SDP. A client, who supports the "alt", "alt-default-id" and "alt-group" attributes, can now select the most suitable alternative by using the control URLs corresponding to the selected alternatives in the RTSP SETUP message. The server sets up the selected alternatives and the client starts playing them. If the client is unaware of the attributes, they will be ignored. The result will be that the client uses the default "a=control" URLs at setup and receives the default alternatives.

EXAMPLE 7:
```
v=0
o=ericsson_user 1 1 IN IP4 130.240.188.69
s=A basic audio and video presentation
c=IN IP4 0.0.0.0
b=AS:56
b=TIAS: 47700
a=maxprate:25
a=control:*
a=range:npt=0-150.2
a=alt-group:BW:AS:28=1,3;56=1,4;60=2,4;120=2,5
a=alt-group:BW:TIAS:21300_20=1,3;47700_25=1,4;43880_25=2,4;112480_35=2,5
t=0 0
m=audio 0 RTP/AVP 97
b=AS:12
b=TIAS:8500
a=maxprate:10
a=rtpmap:97 AMR/8000
a=control:trackID=1
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
a=alt-default-id:1
a=alt:2:b=AS:16
a=alt:2:b=TIAS:12680
```

```
a=alt:2:a=control:trackID=2
m=video 0 RTP/AVP 98
b=AS:44
b=TIAS:39200
a=maxprate:15
a=rtpmap:98 MP4V-ES/90000
a=control:trackID=4
a=fmtp:98 profile-level-id=8; config=01010000012000884006682C2090A21F
a=range:npt=0-150.2
a=X-initpredecbufperiod:98000
a=alt-default-id:4
a=alt:3:b=AS:16
a=alt:3:b=TIAS:12800
a=alt:3:a=maxprate:10
a=alt:3:a=control:trackID=3
a=alt:3:a=X-initpredecbufperiod:48000
a=alt:5:b=AS:104
a=alt:5:b=TIAS:99800
a=alt:5:a=maxprate:25
a=alt:5:a=control:trackID=5
a=alt:5:a=X-initpredecbufperiod:150000
```

The above example has 5 alternatives, 2 for audio and 3 for video. That would allow for a total of six combinations between audio and video. However, the grouping attribute in this example recommends that only 4 of these combinations be used. The equivalent SDP for the default alternatives (alternatives 1 and 4) with a total session bitrate of 56 kbps follows:

EXAMPLE 8:

```
v=0
o=ericsson_user 1 1 IN IP4 130.240.188.69
s=Ericsson commercial
c=IN IP4 0.0.0.0
b=AS:56
b=TIAS: 47700
a=maxprate:25
a=control:*
a=range:npt=0-150.2
t=0 0
m=audio 0 RTP/AVP 97
b=AS:12
b=TIAS:8500
a=maxprate:10
a=rtpmap:97 AMR/8000
a=control:trackID=1
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
m=video 0 RTP/AVP 98
b=AS:44
b=TIAS:39200
a=maxprate:15
a=rtpmap:98 MP4V-ES/90000
a=control:trackID=4
a=fmtp:98 profile-level-id=8; config=01010000012000884006682C2090A21F
a=range:npt=0-150.2
a=X-initpredecbufperiod:98000
```

The equivalent SDP for the 28 kbps total session bitrate (alternatives 1 and 3) is:

EXAMPLE 9:

```
v=0
o=ericsson_user 1 1 IN IP4 130.240.188.69
s=A basic audio and video presentation
c=IN IP4 0.0.0.0
b=AS:28
```

```
b=TIAS:21300
a=maxprate:20
a=control:*
a=range:npt=0-150.2
t=0 0
m=audio 0 RTP/AVP 97
b=AS:12
b=TIAS:8500
a=maxprate:10
a=rtpmap:97 AMR/8000
a=control:trackID=1
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
m=video 0 RTP/AVP 98
b=AS:16
b=TIAS:12800
a=maxprate:10
a=rtpmap:98 MP4V-ES/90000
a=control:trackID=3
a=fmtp:98 profile-level-id=8; config=01010000012000884006682C2090A21F
a=range:npt=0-150.2
a=X-initpredecbufperiod:48000
```

The equivalent SDP for the grouping with a 120 kbps total session bandwidth (alternatives 2 and 5):

EXAMPLE 10:

```
v=0
o=ericsson_user 1 1 IN IP4 130.240.188.69
s=A basic audio and video presentation
c=IN IP4 0.0.0.0
b=AS:120
b=TIAS: 112480
a=maxprate:35
a=control:*
a=range:npt=0-150.2
t=0 0
m=audio 0 RTP/AVP 97
b=AS:16
b=TIAS:12680
a=maxprate:10
a=rtpmap:97 AMR/8000
a=control:trackID=2
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
m=video 0 RTP/AVP 98
b=AS:104
b=TIAS:99800
a=maxprate:25
a=rtpmap:98 MP4V-ES/90000
a=control:trackID=5
a=fmtp:98 profile-level-id=8; config=01010000012000884006682C2090A21F
a=range:npt=0-150.2
a=X-initpredecbufperiod:150000
```

The recommendation for a session with a total bitrate of 60 kbps is as easily formed. A client will use the received SDP and, as an example available bandwidth, to chose which alternatives to set up. If the client only has 32 kbps it selects the media alternatives 1 and 3, which use 28 kbps. The client sets this up by sending two normal RTSP requests using the control URLs from the chosen alternatives.

The audio SETUP request for the default (i.e. 56 kbps in the example above) looks like this:

EXAMPLE 11: SETUP rtsp://media.example.com/examples/3G_systems.3gp/trackID=1 RTSP/1.0

CSeq: 2
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457

The response from the server would be:

EXAMPLE 12: RTSP/1.0 200 OK
CSeq: 2
Session: jEs.EdXCSKpB
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457;server_port=4002-4003;ssrc=5199dcb1

Also the video is added to the RTSP session under aggregated control:

EXAMPLE 13: SETUP rtsp://media.example.com/examples/3G_systems.3gp/trackID=3 RTSP/1.0
CSeq: 3
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459
Session: jEs.EdXCSKpB

And the response would be:

EXAMPLE 14: RTSP/1.0 200 OK
CSeq: 3
Session: jEs.EdXCSKpB
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459;server_port=4004-4005;ssrc=ae75904f

Had the client had more available bandwidth it could have set up another pair of alternatives in order to get better quality. The only change had been the RTSP URLs that had pointed at other media streams. For example the 120 kbps version would have been received if the audio SETUP request had used:

EXAMPLE 15: rtsp://media.example.com/examples/3G_systems.3gp/trackID=2

and the video request

EXAMPLE 16: rtsp://media.example.com/examples/3G_systems.3gp/trackID=5

The following example shows an SDP file that contains asset information, defined in Clause 5.3.3.7.

EXAMPLE 17:
```
v=0
o=ghost 2890844526 2890842807 IN IP4 192.168.10.10
s=3GPP Unicast SDP Example
i=Example of Unicast SDP file
u=http://www.infoserver.com/ae600
e=ghost@mailserver.com
c=IN IP4 0.0.0.0
t=0 0
a=range:npt=0-45.678
a=3GPP-Asset-Information: {url="http://www.movie-database.com/title/thismovieinfo.xhtml"}
a=3GPP-Asset-Information: {Title=MjhDRTA2NzI},{Copyright=Mjc0MkUwMUVGNDE2}
m=video 1024 RTP/AVP 96
b=AS:128
b=TIAS:118400
a=maxprate:30
a=rtpmap:96 H263-2000/90000
a=fmtp:96 profile=3;level=10
a=control:rtsp://mediaserver.com/movie.3gp/trackID=1
a=framesize:96 176-144
```

A.2 RTSP

A.2.1 General

Clause 5.3.2 of the present document defines the required RTSP support in PSS clients and servers by making references to Appendix D of [5]. It also defines the RTSP header fields that are specific to PSS. The current clause gives an informative overview of these methods (see Table A.2) and headers (see Table A.3). Note that this overview does not replace the information in Appendix D of [5] and Clause 5.3.2 of the present document, which must be consulted for a full implementation of RTSP in PSS. Two examples of RTSP sessions are also given.

Table A.2: Overview of the RTSP method support in PSS

Method	Requirement for a minimal on-demand playback client according to [5].	Requirement for a PSS client according to the present document.	Requirement for a minimal on-demand playback server according to [5].	Requirement for a PSS server according to the present document.
OPTIONS	O	O	Respond	Respond
REDIRECT	Respond	Respond	O	O
DESCRIBE	O	Generate	O	Respond
SETUP	Generate	Generate	Respond	Respond
PLAY	Generate	Generate	Respond	Respond
PAUSE	Generate	Generate	Respond	Respond
TEARDOWN	Generate	Generate	Respond	Respond
SET_PARAMETER	O	O	O	O

NOTE 1: O = Support is optional
NOTE 2: 'Generate' means that the client/server is required to generate the request where applicable.
NOTE 3: 'Respond' means that the client/server is required to properly respond to the request.

Table A.3: Overview of the RTSP header support in PSS

Header	Requirement for a minimal on-demand playback client according to [5].	Requirement for a PSS client according to the present document.	Requirement for a minimal on-demand playback server according to [5].	Requirement for a PSS server according to the present document.
Bandwidth	O	O	O	O
Connection	include/understand	include/understand	include/understand	include/understand
Content-Encoding	understand	understand	include	include
Content-Language	understand	understand	include	include
Content-Length	understand	understand	include	include
Content-Type	understand	understand	include	include
CSeq	include/understand	include/understand	include/understand	include/understand
Date	include	include	include	include
Location	understand	understand	O	O
Public	O	O	include	include
Range	O	include/understand	understand	include/understand
Require	O	O	understand	understand
RTP-Info	understand	understand	include	include
Server[4]	O	O	O	O
Session	include	include	understand	understand
Timestamp	O	O	include/understand	include/understand
Transport	include/understand	include/understand	include/understand	include/understand
Unsupported	include	include	include	include
User-Agent[4]	O	O	O	O
3GPP-Adaptation	N/A	O	N/A	O
3GPP-Link-Char	N/A	O	N/A	O
3GPP-QoE-Metrics	N/A	O	N/A	O

NOTE 1: O = Support is optional
NOTE 2: 'include' means that the client/server is required to include the header in a request or response where applicable.
NOTE 3: 'understand' means that the client/server is required to be able to respond properly if the header is received in a request or response.
NOTE 4: According to [5] the "Server" and 'User-Agent' headers are not strictly required for a minimal RTSP implementation, although it is highly recommended that they are included with responses and requests. The same applies to PSS servers and clients according to the present document.

The example below is intended to give some more understanding of how RTSP and SDP are used within the 3GPP PSS. The example assumes that the streaming client has the RTSP URL to a presentation consisting of an H.263 video sequence and AMR speech. RTSP messages sent from the client to the server are in **bold** and messages from the server to the client in *italic*. In the example the server provides aggregate control of the two streams.

EXAMPLE 1:

DESCRIBE rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 1
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 1
Content-Type: application/sdp
Content-Length: 435

v=0
o=- 950814089 950814089 IN IP4 144.132.134.67
s=Example of aggregate control of AMR speech and H.263 video
e=foo@bar.com

c=IN IP4 0.0.0.0
b=AS:77
b=TIAS:69880
t=0 0
a=range:npt=0-59.3478
a=control:*
a=maxprate:20

m=audio 0 RTP/AVP 97
b=AS:13
b=TIAS:10680
b=RR:350
b=RS:300
a=maxprate:5
a=rtpmap:97 AMR/8000
a=fmtp:97
a=maxptime:200
a=control:streamID=0
m=video 0 RTP/AVP 98
b=AS:64
b=TIAS:59200
b=RR:2000
b=RS:1200
a=maxprate:15
a=rtpmap:98 H263-2000/90000
a=fmtp:98 profile=3;level=10
a=control:streamID=1

SETUP rtsp://mediaserver.com/movie.test/streamID=0 RTSP/1.0
CSeq: 2
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 2
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457; server_port=5678-5679
Session: dfhyrio90llk

SETUP rtsp://mediaserver.com/movie.test/streamID=1 RTSP/1.0
CSeq: 3
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 3
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459; server_port=5680-5681
Session: dfhyrio90llk

PLAY rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 4
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 4
Session: dfhyrio90llk
Range: npt=0-
RTP-Info: url= rtsp://mediaserver.com/movie.test/streamID=0; seq=9900;rtptime=4470048,
url= rtsp://mediaserver.com/movie.test/streamID=1; seq=1004;rtptime=1070549

NOTE: Headers can be folded onto multiple lines if the continuation line begins with a space or horizontal tab. For more information, see RFC2616 [17].

The user watches the movie for 20 seconds and then decides to jump to 10 seconds before the end…

PAUSE rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 5
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

PLAY rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 6
Range: npt=50-59.3478
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 5
Session: dfhyrio90llk

RTSP/1.0 200 OK
CSeq: 6
Session: dfhyrio90llk
Range: npt=50-59.3478
RTP-Info: url= rtsp://mediaserver.com/movie.test/streamID=0;
seq=39900;rtptime=44470648,
url= rtsp://mediaserver.com/movie.test/streamID=1;
seq=31004;rtptime=41090349

After the movie is over the client issues a TEARDOWN to end the session…

TEARDOWN rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 7
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
Cseq: 7
Session: dfhyrio90llk
Connection: close

The example below contains a complete RTSP signalling for session set-up with rate adaptation support, where the client buffer feedback functionality is initialised and used. To allow the server to know that a client supports the buffer feedback formats and signalling, the client includes a link to its UAProf description in its RTSP DESCRIBE request.

EXAMPLE 2:

DESCRIBE rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 1
User-Agent: TheStreamClient/1.1b2
x-wap-profile: "http://uaprof.example.com/products/TheStreamClient1.1b2"

```
RTSP/1.0 200 OK
CSeq: 1Date: 20 Aug 2003 15:35:06 GMT
Content-Base: rtsp://mediaserver.com/movie.test/
Content-Type: application/sdp
Content-Length: 500

v=0
o=- 950814089 950814089 IN IP4 144.132.134.67
s=Example of aggregate control of AMR speech and H.263 video
e=foo@bar.com
c=IN IP4 0.0.0.0
b=AS:77
b=TIAS:69880
t=0 0
a=maxprate:20
a=range:npt=0-59.3478
a=control:*
m=audio 0 RTP/AVP 97
b=AS:13
b=TIAS:10680
b=RR:350
b=RS:300
a=maxprate:5
a=rtpmap:97 AMR/8000
a=fmtp:97 octet-align=1
a=control: streamID=0
a=3GPP-Adaptation-Support:2
m=video 0 RTP/AVP 98
b=AS:64
b=TIAS:59200
b=RR:2000
b=RS:1200
a=maxprate:15
a=rtpmap:98 H263-2000/90000
a=fmtp:98 profile=3;level=10
a=control: streamID=1
a=3GPP-Adaptation-Support:1
```

SETUP rtsp://mediaserver.com/movie.test/streamID=0 RTSP/1.0
CSeq: 2
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457
User-Agent: TheStreamClient/1.1b2
3GPP-Adaptation: url="rtsp://mediaserver.com/movie.test/streamID=0";size=14500;target-time=5000

```
RTSP/1.0 200 OK
CSeq: 2
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457;server_port=5678-5679;ssrc=A432F9B1
Session: dfhyrio90llk
3GPP-Adaptation: url="rtsp://mediaserver.com/movie.test/streamID=0";size=14500;target-time=5000
```

SETUP rtsp://mediaserver.com/movie.test/streamID=1 RTSP/1.0
CSeq: 3
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2
3GPP-Adaptation: url="rtsp://mediaserver.com/movie.test/streamID=1";size=35000;target-time=5000

RTSP/1.0 200 OK
CSeq: 3
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459; server_port=5680-5681; ssrc=4D23AE29
Session: dfhyrio90llk
3GPP-Adaptation: url=" rtsp://mediaserver.com/movie.test/streamID=1";size=35000;target-time=5000

PLAY rtsp://mediaserver.com/movie.test/ RTSP/1.0
CSeq: 4
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 4
Session: dfhyrio90llk
Range: npt=0-
RTP-Info: url= rtsp://mediaserver.com/movie.test/streamID=0; seq=9900;rtptime=4470048, url= rtsp://mediaserver.com/movie.test/streamID=1; seq=1004;rtptime=1070549

If the client desires to change the target buffer protection time during the session, it can signal a new value to the server by means of an RTSP SET_PARAMETER request.

SET_PARAMETER rtsp://mediaserver.com/movie.test/ RTSP/1.0
CSeq: 8
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2
3GPP-Adaptation: url="rtsp://mediaserver.com/movie.test/streamID=0";target-time=7000,url="rtsp://mediaserver.com/movie.test/streamID=1";target-time=7000

RTSP/1.0 200 OK
CSeq: 8
Session: dfhyrio90llk
3GPP-Adaptation: url="rtsp://mediaserver.com/movie.test/streamID=0";target-time=7000,url="rtsp://mediaserver.com/movie.test/streamID=1";target-time=7000

A.2.2 Implementation guidelines

A.2.2.1 Usage of persistent TCP

Considering the potentially long round-trip-delays in a packet switched streaming service over UMTS it is important to keep the number of messages exchanged between a server and a client low. The number of requests and responses exchanged is one of the factors that will determine how long it takes from the time that a user initiates PSS until the streams starts playing in a client.

RTSP methods are sent over either TCP or UDP for IP. Both client and server shall support RTSP over TCP whereas RTSP over UDP is optional. For TCP the connection can be persistent or non-persistent. A persistent connection is used for several RTSP request/response pairs whereas one connection is used per RTSP request/response pair for the non-persistent connection. In the non-persistent case each connection will start with the three-way handshake (SYN, ACK, SYN) before the RTSP request can be sent. This will increase the time for the message to be sent by one round trip delay.

For these reasons it is recommended that 3GPP PSS clients should use a persistent TCP connection, at least for the initial RTSP methods until media starts streaming.

A.2.2.2 Detecting link aliveness

In the wireless environment, connection may be lost due to fading, shadowing, loss of battery power, or turning off the terminal even though the PSS session is active. In order for the server to be able to detect the client"s aliveness, the PSS client should send 'wellness' information to the PSS server for a defined interval as described in the RFC2326. There are several ways for detecting link aliveness described in the RFC2326, however, the client should be careful about issuing 'PLAY method without Range header field' too close to the end of the streams, because it may conflict with pipelined PLAY requests. Below is the list of recommended 'wellness' information for the PSS clients and servers in a prioritised order.

1. RTCP

2. OPTIONS method with Session header field

NOTE: Both servers and clients can initiate this OPTIONS method.

The client should send the same wellness information in "Ready" state as in "Playing" and "Recording" states, and the server should detect the same client"s wellness information in "Ready" state as in "Playing" and "Recording" states. In particular, the same link aliveness mechanism should be managed following a "PAUSE" request and response.

A.3 RTP

A.3.1 General

Void.

A.3.2 Implementation guidelines

A.3.2.1 Maximum RTP packet size

The RFC 3550 (RTP) [9] does not impose a maximum size on RTP packets. However, when RTP packets are sent over the radio link of a 3GPP PSS system there is an advantage in limiting the maximum size of RTP packets.

Two types of bearers can be envisioned for streaming using either acknowledged mode (AM) or unacknowledged mode (UM) RLC. The AM uses retransmissions over the radio link whereas the UM does not. In UM mode large RTP packets are more susceptible to losses over the radio link compared to small RTP packets since the loss of a segment may result in the loss of the whole packet. On the other hand in AM mode large RTP packets will result in larger delay jitter compared to small packets as there is a larger chance that more segments have to be retransmitted.

For these reasons it is recommended that the maximum size of RTP packets should be limited in size taking into account the wireless link. This will decrease the RTP packet loss rate particularly for RLC in UM. For RLC in AM the delay jitter will be reduced permitting the client to use a smaller receiving buffer. It should also be noted that too small RTP packets could result in too much overhead if IP/UDP/RTP header compression is not applied or unnecessary load at the streaming server.

In the case of transporting video in the payload of RTP packets it may be that a video frame is split into more than one RTP packet in order not to produce too large RTP packets. Then, to be able to decode packets following a lost packet in the same video frame, it is recommended that synchronisation information be inserted at the start of such RTP packets.

For H.263 this implies the use of GOBs with non-empty GOB headers and in the case of MPEG-4 video the use of video packets (resynchronisation markers). If the optional Slice Structured mode (Annex K) of H.263 is in use, GOBs are replaced by slices.

A.3.2.2 Sequence number and timestamp in the presence of NPT jump

The description below is intended to give more understanding of how RTP sequence number and timestamp are specified within the 3GPP PSS in the presence of NPT jumps. The jump happens when a client sends a PLAY request to skip media.

The RFC 2326 (RTSP) [5] specifies that both RTP sequence numbers and RTP timestamps must be continuous and monotonic across jumps of NPT. Thus when a server receives a request for a skip of the media that causes a jump of NPT, it shall specify RTP sequence numbers and RTP timestamps continuously and monotonically across the skip of the media to conform to the RTSP specification. Also, the server may respond with "seq" in the RTP-Info field if this parameter is known at the time of issuing the response.

A.3.2.3 RTCP transmission interval

In RTP [9] when using the basic RTP profile AVP [10], Section 6.2 of [9] defines rules for the calculation of the interval between the sending of two consecutive RTCP packets, i.e. the RTCP transmission interval. These rules consist of two steps:

- Step 1: an algorithm that calculates a transmission interval from parameters such as the RTCP bandwidth defined in section 5.3.3.1 and the average RTCP packet size. This algorithm is described in [9], with example code in annex A.7.
- Step 2: Taking the maximum of the transmission interval computed in step 1 and a mandatory fixed minimum RTCP transmission interval. The RTP/RTCP specification [9] gives a recommendation that the minimum interval is set to 5 seconds, but it may be scaled to other values in unicast sessions for all participants (SSRCs), see section 6.2 of [9] for further details. For PSS and the AVP profile the minimum interval shall be 5 seconds.

NOTE: The algorithm in Annex A.7 of [9] must be accordingly modified to enable usage of the explicit bandwidth values given for the RTCP bandwidth, as provided by the SDP bandwidth modifiers (RR and RS) that shall be used by PSS according to clause 5.3.3.1.

Implementations conforming to this TS shall perform step 1 and may perform step 2. All other algorithms and rules of [9] stay valid and shall be followed. Please note that the processing described in [9] include a randomisation with an equally distributed random function resulting in a value somewhere between 0.5 to 1.5 times the calculated value prior to further scaling with a factor of 1/(e-1.5). Those RTCP intervals either can be compared as the average value or as the maximum interval.

The rules defined in RTP [9] and AVP [10] are updated by the AVPF profile [57]. The new rules remove the minimum transmission interval rule. It also provides SDP signalling that allows the server to configure the RTCP behaviour. When using the AVPF profile the PSS client and server shall send RTCP according to the rules in [57] and comply with the signalled parameters.

Below are formulas for calculating the maximal RTCP interval for given input parameters. Normally the RTCP packets will be sent with smaller intervals. The formulas below have been reduced as much as possible and utilize the rules resulting in the largest interval. The formulas are not a replacement for implementing the algorithm in any stack, as some of the input values are dynamic and will change during a session.

Variables:

RSv:	The RTCP bandwidth in bits/s assigned to active data senders
RRv:	The RTCP bandwidth in bits/s assigned to data receiver only.
members:	The total number of participants (SSRCs) in the session.
avg_rtcp_size:	The average RTCP packet size in bytes.
min_rtcp_interval:	The minimum RTCP transmission interval in seconds.
t_rr_interval:	The minimum reporting interval in seconds when in regular RTCP mode for AVPF.

The calculation for the AVP profile:

x = 1.5 * max((avg_rtcp_size * 8 * members / min(RSv, RRv)), min_rtcp_interval) / 1.21828

The calculation for the AVPF profile:

x =1.5 * max(2*(avg_rtcp_size * 8 * members / min(RSv, RRv)) / 1.21828, t_rr_interval)

The above formulas are valid for both a PSS server and a PSS client, and either side can compute the maximum RTCP interval of either of the two sides. For example, the PSS server can compute the maximum RTCP transmission interval for the RTCP packets received by the PSS client just by replacing the expression min(RSv, RRv) with RRv in the formula.

When using the AVPF profile the sending of RTCP reports is governed by the AVPF mode in use, the RTCP bandwidth, the average RTCP packet size and possibly the minimal reporting interval (t_rr_interval). In AVPF the RTCP sender will work in regular reporting mode, unless there are any events to report on. This means that the normal bandwidth limitation rule is used, possibly combined with suppression based on the t_rr_interval variable. The t_rr_interval variable can be set using signalling in SDP with the "trr-int" parameter. Also, due to the transitions between early RTCP mode and the regular reporting mode the reporting can be delayed a complete regular reporting interval. The other modes will all send RTCP at least as often as for the transition between early and regular mode.

A.3.2.4 Timestamp handling after PAUSE/PLAY requests

The description below intends to clarify how RTP timestamps are specified within the 3GPP PSS when a client sends a PLAY request following a PAUSE request. The RTP timestamp space must be continuous along time during a session and then reflect the actual time elapsed since the beginning of the session. A server must reflect the actual time interval elapsed between the last RTP packets sent before the reception of the PAUSE request and the first RTP packets sent after the reception of the PLAY request in the RTP timestamp. A client will need to compute the mapping between NPT time and RTP timestamp each time it receives a PLAY response for on-demand content. This means that a client must be able to cope with any gap in RTP timestamps after a PLAY request.

The PLAY request can include a Range header if the client wants to seek backward or forward in the media, or without a Range header if the client only wants to resume the paused session.

Example:
In this example Client C plays a media file from Server S. RTP timestamp rate in this example is 1000Hz for clarity.

```
C -> S:    PLAY rtsp://example.com/mediastream RTSP/1.0
    CSeq: 2
    Session: 123456
    Range: npt=1.125-


S -> C:    RTSP/1.0 200 OK
    CSeq: 2
    Session: 123456
    Range: npt=1.120-
    RTP-Info: url=rtsp://example.com/mediastream;seq=1000;rtptime=5000


S -> C:    RTP packet - seq = 1000 - rtptime = 5000 - corresponding media time (NPT time) =  1120ms
S -> C:    RTP packet - seq = 1001 - rtptime = 5040 - corresponding media time (NPT time) =  1160ms
S -> C:    RTP packet - seq = 1002 - rtptime = 5080 - corresponding media time (NPT time) =  1200ms
S -> C:    RTP packet - seq = 1003 - rtptime = 5120 - corresponding media time (NPT time) =  1240ms


C -> S: PAUSE rtsp://example.com/mediastream RTSP/1.0
    CSeq: 3
    Session: 123456


S -> C:    RTSP/1.0 200 OK
    CSeq: 3
```

Session: 123456

[10 seconds elapsed]

C -> S: PLAY rtsp://example.com/mediastream RTSP/1.0
CSeq: 4
Session: 123456

S -> C: RTSP/1.0 200 OK
CSeq: 4
Session: 123456
Range: npt=1.280-
RTP-Info: url=rtsp://example.com/mediastream;seq=1004;rtptime=15160

S -> C: RTP packet - seq = 1004 - rtptime = 15160 - corresponding media time (NPT time) = 1280ms
S -> C: RTP packet - seq = 1005 - rtptime = 15200 - corresponding media time (NPT time) = 1320ms
S -> C: RTP packet - seq = 1006 - rtptime = 15240 - corresponding media time (NPT time) = 1360ms

C -> S: PAUSE rtsp://example.com/mediastream RTSP/1.0
CSeq: 5
Session: 123456

S -> C: RTSP/1.0 200 OK
CSeq: 5
Session: 123456

C -> S: PLAY rtsp://example.com/mediastream RTSP/1.0
CSeq: 6
Session: 123456
Range: npt=0.5-

[55 milliseconds elapsed during request processing]

S -> C: RTSP/1.0 200 OK
CSeq: 6
Session: 123456
Range: npt=0.480-
RTP-Info: url=rtsp://example.com/mediastream;seq=1007;rtptime=15295

S -> C: RTP packet - seq = 1007 - rtptime = 15295 - corresponding media time (NPT time) = 480ms
S -> C: RTP packet - seq = 1008 - rtptime = 15335 - corresponding media time (NPT time) = 520ms
S -> C: RTP packet - seq = 1009 - rtptime = 15375 - corresponding media time (NPT time) = 560ms

A.3.3 Examples of RTCP APP packets for client buffer feedback

Example 1: The RTCP Receiver Report and NADU packet while having a number of packets for a single source in the receiver buffer and signalling the playout delay for the next unit to be decoded.

RTCP Receiver Report:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|    RC   |   PT=RR=201   |             length = 7       |
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    SSRC of packet sender = 0x324FE239         |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                SSRC_1 (SSRC of first source) = 0x4D23AE29     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| fraction lost |       cumulative number of packets lost       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| extended highest sequence number received = 0x00000551 (1361) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      interarrival jitter                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         last SR (LSR)                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   delay since last SR (DLSR)                  |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

APP packet:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|subtype=0|   PT=APP=204  |             length = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Client SSRC = 0x324FE239                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         name = "PSS0"                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Server SSRC = 0x4D23AE29                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Playout Delay = 300       |             NSN = 1323        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Reserved     | NUN = 2 |             FBS = 292               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

From the above compound RTCP packet, the server is able to derive all the ADUs that are in the receiver buffer by looking up all the ADUs it has sent which follow in decoding the second unit of packet SN 1323 and which were sent up to packet 1361. The total buffer size is 35000 bytes as indicated during the RTSP session setup (see rate-adaptation example in clause A.2.1). The available free space in the buffer is report as 292 64-byte blocks, which equals 18688 bytes of free buffer space.

The server is able to measure the time difference between the next ADU to be decoded and the next ADU it will send by comparing the decoding times of these units. Depending on this value, it is able to adapt using e.g. bitstream switching or bitstream thinning.

If the receiver had chosen not to signal the playout delay of the oldest packet, the receiver would have sent instead the reserved value 0xFFFF for the playout delay field.

Example 2: Reporting an empty buffer.

In the case a client has played out all packets for a SSRC that has been received and would send out a RTCP receiver report according to the one in example 1, the NADU packet would carry an NSN value of 1362. This results in that the calculation of the number of packets becomes 0 (1361-1362+1). As the buffer is empty, the playout delay is not defined and the receiver should use the reserved value 0xFFFF for this field.

A.4 Capability exchange

A.4.1 Overview

Clause A.4 provides detailed information about the structure and exchange of device capability descriptions for the PSS. It complements the normative part contained in clause 5.2 of the present document.

The functionality is sometimes referred to as capability exchange. Capability exchange in PSS uses the CC/PP [39] framework and reuse parts of the CC/PP application UAProf [40].

To facilitate server-side content negotiation for streaming, the PSS server needs to have access to a description of the specific capabilities of the mobile terminal, i.e. the device capability description. The device capability description contains a number of attributes. During the set-up of a streaming session the PSS server can use the description to provide the mobile terminal with the correct type of multimedia content. Concretely, it is envisaged that servers use information about the capabilities of the mobile terminal to decide which stream(s) to provision to the connecting terminal. For instance, the server could compare the requirements on the mobile terminal for multiple available variants of a stream with the actual capabilities of the connecting terminal to determine the best-suited stream(s) for that particular terminal. A similar mechanism could also be used for other types of content.

A device capability description contains a number of device capability attributes. In the present document they are referred to as just attributes. The current version of PSS does not include a definition of any specific user preference attributes. Therefore we use the term device capability description. However, it should be noted that even though no specific user preference attributes are included, simple tailoring to the preferences of the user could be achieved by temporarily overrides of the available attributes. E.g. if the user for a particular session only would like to receive mono sound even though the terminal is capable of stereo, this can be accomplished by providing an override for the "AudioChannels" attribute. It should also be noted that the extension mechanism defined would enable an easy introduction of specific user preference attributes in the device capability description if needed.

The term device capability profile or profile is sometimes used instead of device capability description to describe a description of device capabilities and/or user preferences. The three terms are used interchangeably in the present document.

Figure A.1 illustrates how capability exchange in PSS is performed. In the simplest case the mobile terminal informs the PSS server(s) about its identity so that the latter can retrieve the correct device capability profile(s) from the device profile server(s). For this purpose, the mobile terminal adds one or several URLs to RTSP and/or HTTP protocol data units that it sends to the PSS server(s). These URLs point to locations on one or several device profile servers from where the PSS server should retrieve the device capability profiles. This list of URLs is encapsulated in RTSP and HTTP protocol data units using additional header field(s). The list of URLs is denoted URLdesc. The mobile terminal may supplement the URLdesc with extra attributes or overrides for attributes already defined in the profile(s) located at URLdesc. This information is denoted Profdiff. As URLdesc, Profdiff is encapsulated in RTSP and HTTP protocol data units using additional header field(s).

The device profile server in Figure A.1 is the logical entity that stores the device capability profiles. The profile needed for a certain request from a mobile terminal may be stored on one or several such servers. A terminal manufacturer or a software vendor could maintain a device profile server to provide device capability profiles for its products. It would also be possible for an operator to manage a device profile server for its subscribers and then e.g. enable the subscriber to make user specific updates to the profiles. The device profile server provides device capability profiles to the PSS server on request.

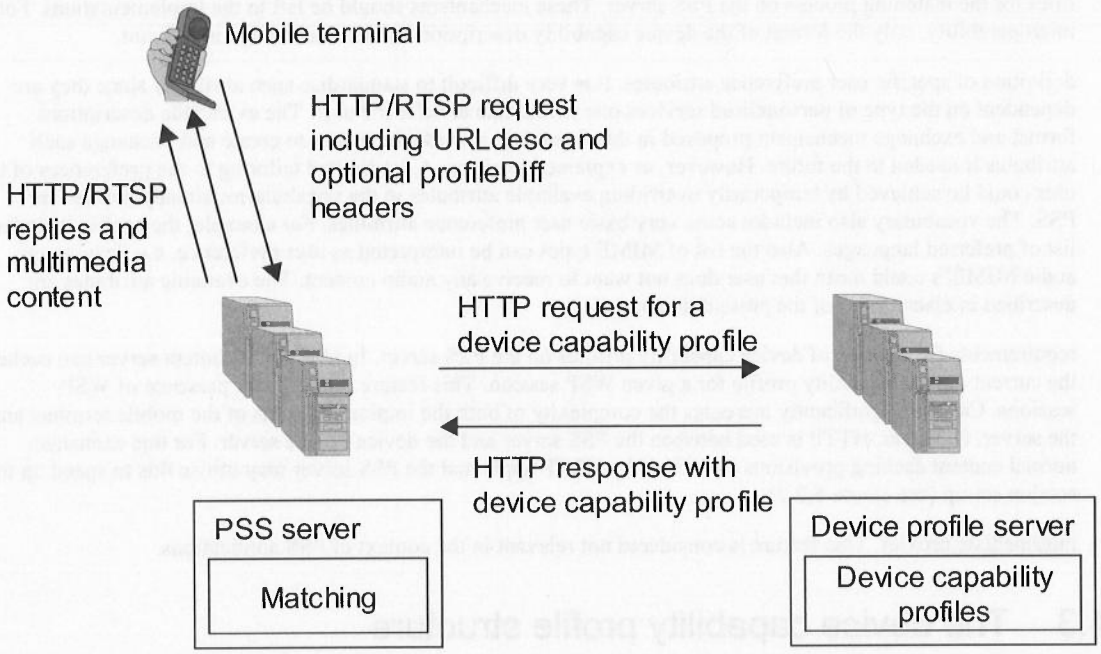


Figure A.1: Functional components in PSS capability exchange

The PSS server is the logical entity that provides multimedia streams and other, static content (e.g. SMIL documents, images, and graphics) to the mobile terminal (see Figure A.1). A PSS application might involve multiple PSS servers, e.g. separate servers for multimedia streams and for static content. A PSS server handles the matching process. Matching is a process that takes place in the PSS servers (see Figure A.1). The device capability profile is compared with the content descriptions at the server and the best fit is delivered to the client.

A.4.2 Scope of the specification

The following bullet list describes what is considered to be within the scope of the specification for capability exchange in PSS.

- Definition of the structure for the device capability profiles, see clause A.4.3.
- Definition of the CC/PP vocabularies, see clause A.4.4.
  - Reference to a set of device capability attributes for multimedia content retrieval applications that have already been defined by UAProf [40]. The purpose of this reference is to point out which attributes are useful for the PSS application.
  - Definition of a set of device capability attributes specifically for PSS applications that are missing in UAProf.
- It is important to define an extension mechanism to easily add attributes since it is not possible to cover all attributes from the beginning. The extension mechanism is described in clause A.4.5.
- The structure of URLdesc, Profdiff and their interchange is described in clause A.4.6.
- Protocols for the interchange of device capability profiles between the PSS server and the device profile server is defined in clause 5.2.7.

The specification does not include:

- rules for the matching process on the PSS server. These mechanisms should be left to the implementations. For interoperability, only the format of the device capability description and its interchange is relevant.

- definition of specific user preference attributes. It is very difficult to standardise such attributes since they are dependent on the type of personalised services one would like to offer the user. The extensible descriptions format and exchange mechanism proposed in this document provide the means to create and exchange such attributes if needed in the future. However, as explained in clause A.4.1 limited tailoring to the preferences of the user could be achieved by temporarily overriding available attributes in the vocabularies already defined for PSS. The vocabulary also includes some very basic user preference attributes. For example, the profile includes a list of preferred languages. Also the list of MIME types can be interpreted as user preference, e.g. leaving out audio MIME"s could mean that user does not want to receive any audio content. The available attributes are described in clause 5.2.3 of the present document.

- requirements for caching of device capability profiles on the PSS server. In UAProf, a content server can cache the current device capability profile for a given WSP session. This feature relies on the presence of WSP sessions. Caching significantly increases the complexity of both the implementations of the mobile terminal and the server. However, HTTP is used between the PSS server and the device profile server. For this exchange, normal content caching provisions as defined by HTTP apply and the PSS server may utilise this to speed up the session set-up (see clause 5.2.7)

- intermediate proxies. This feature is considered not relevant in the context of PSS applications.

A.4.3 The device capability profile structure

A device capability profile is a description of the capabilities of the device and possibly also the preferences of the user of that device. It can be used to guide the adaptation of content presented to the device. A device capability profile for PSS is an RDF [41] document that follows the structure of the CC/PP framework [39] and the CC/PP application UAProf [40]. The terminology of CC/PP is used in this text and therefore briefly described here.

Attributes are used for specifying the device capabilities and user preferences. A set of attribute names, permissible values and semantics constitute a CC/PP vocabulary. An RDF schema defines a vocabulary. The syntax of the attributes is defined in the schema but also, to some extent, the semantics. A profile is an instance of a schema and contains one or more attributes from the vocabulary. Attributes in a schema are divided into components distinguished by attribute characteristics. In the CC/PP specification it is anticipated that different applications will use different vocabularies. According to the CC/PP framework a hypothetical profile might look like Figure A.2. A further illustration of how a profile might look like is given in the example in clause A.4.7.

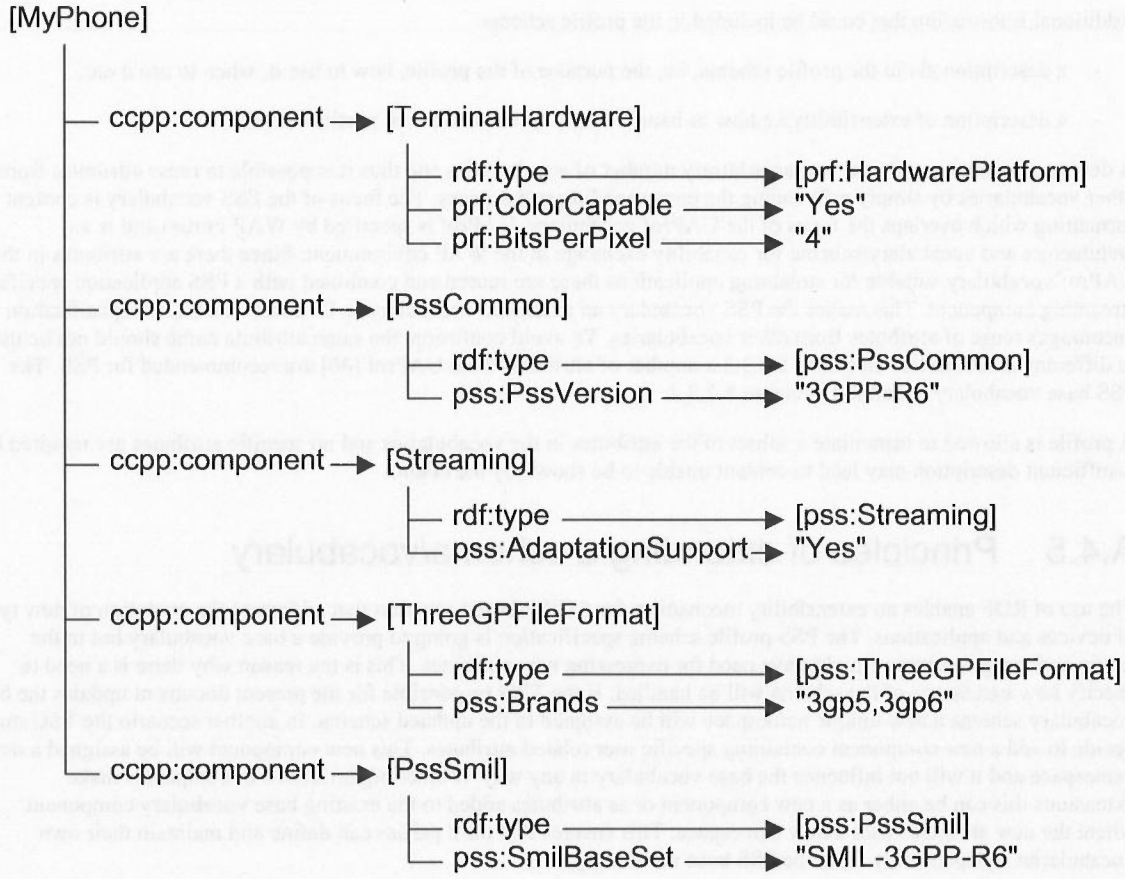


Figure A.2: Illustration of the profile structure

A CC/PP schema is extended through the introduction of new attribute vocabularies and a device capability profile can use attributes drawn from an arbitrary number of different vocabularies. Each vocabulary is associated with a unique XML namespace. This mechanism makes it possible to reuse attributes from other vocabularies. It should be mentioned that the prefix **ccpp** identifies elements of the CCPP namespace (URI http://www.w3.org/2002/11/08-ccpp-ns#), **prf** identifies elements of the UAProf namespace (URI http://www.wapforum.org/profiles/UAPROF/ccppschema-20010330#), **rdf** identifies elements of the RDF namespace (URI http://www.w3.org/1999/02/22-rdf-syntax-ns#) and **pss** identifies elements of the PSS Release-6 namespace. (URI http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#).

Attributes of a component can be included directly or may be specified by a reference to a CC/PP default profile. Resolving a profile that includes a reference to a default profile is time-consuming. When the PSS server receives the profile from a device profile server the final attribute values can not be determined until the default profile has been requested and received. Support for defaults is required by the CC/PP specification [39]. Due to these problems, there is a recommendation made in clause 5.2.6 to not use the CC/PP defaults element in PSS device capability profile documents.

A.4.4 CC/PP Vocabularies

A CC/PP vocabulary shall according to CC/PP and UAProf include:

- an RDF schema for the vocabulary based on the CC/PP schema;
- a description of the semantics/type/resolution rules/sample values for each attribute;

- a unique namespace shall be assigned to each version of the profile schema.

Additional information that could be included in the profile schema:

- a description about the profile schema, i.e. the purpose of the profile, how to use it, when to use it etc;
- a description of extensibility,i.e.how to handle future extensions of the profile schema.

A device capability profile can use an arbitrary number of vocabularies and thus it is possible to reuse attributes from other vocabularies by simply referencing the corresponding namespaces. The focus of the PSS vocabulary is content formatting which overlaps the focus of the UAProf vocabulary. UAProf is specified by WAP Forum and is an architecture and vocabulary/schema for capability exchange in the WAP environment. Since there are attributes in the UAProf vocabulary suitable for streaming applications these are reused and combined with a PSS application specific streaming component. This makes the PSS vocabulary an extension vocabulary to UAProf. The CC/PP specification encourages reuse of attributes from other vocabularies. To avoid confusion, the same attribute name should not be used in different vocabularies. In clause 5.2.3.3 a number of attributes from UAProf [40] are recommended for PSS. The PSS base vocabulary is defined in clause 5.2.3.2.

A profile is allowed to instantiate a subset of the attributes in the vocabularies and no specific attributes are required but insufficient description may lead to content unable to be shown by the client.

A.4.5 Principles of extending a schema/vocabulary

The use of RDF enables an extensibility mechanism for CC/PP-based schemas that addresses the evolution of new types of devices and applications. The PSS profile schema specification is going to provide a base vocabulary but in the future new usage scenarios might have need for expressing new attributes. This is the reason why there is a need to specify how extensions of the schema will be handled. If the TSG responsible for the present document updates the base vocabulary schema a new unique namespace will be assigned to the updated schema. In another scenario the TSG may decide to add a new component containing specific user related attributes. This new component will be assigned a new namespace and it will not influence the base vocabulary in any way. If other organisations or companies make extensions this can be either as a new component or as attributes added to the existing base vocabulary component where the new attributes uses a new namespace. This ensures that third parties can define and maintain their own vocabularies independently from the PSS base vocabulary.

A.4.6 Signalling of profile information between client and server

URLdesc and Profdiff were introduced in clause A.4.1. The URLdesc is a list of URLs that point to locations on device profile servers from where the PSS server retrieves suitable device capability profiles. The Profdiff contains additional capability description information; e.g. overrides for certain attribute values. Both URLdesc and Profdiff are encapsulated in RTSP and HTTP messages using additional header fields. This can be seen in Figure A.1. In clause 9.1 of [40] three new HTTP headers are defined that can be used to implement the desired functionality: "x-wap-profile", "x-wap-profile-diff" and "x-wap-profile-warning". These headers are reused in PSS for both HTTP and RTSP.

- The "x-wap-profile" is a request header that contains a list of absolute URLs to device capability descriptions and profile diff names. The profile diff names correspond to additional profile information in the "x-wap-profile-diff" header.
- The "x-wap-profile-diff" is a request header that contains a subset of a device capability profile.
- The "x-wap-profile-warning" is a response header that contains error codes explaining to what extent the server has been able to match the terminal request.

Clause 5.2.5 of the present document defines this exchange mechanism.

It is left to the mobile terminal to decide when to send x-wap-profile headers. The mobile terminal could send the "x-wap-profile" and "x-wap-profile-diff" headers with each RTSP DESCRIBE and/or with each RTSP SETUP request. Sending them in the RTSP DESCRIBE request is useful for the PSS server to be able to make a better decision which presentation description to provision to the client. Sending the "x-wap-profile" and "x-wap-profile-diff" headers with an HTTP request is useful whenever the mobile terminal requests some multimedia content that will be used in the PSS application. For example it can be sent with the request for a SMIL file and the PSS server can see to it that the mobile terminal receives a SMIL file which is optimised for the particular terminal. Clause 5.2.5 of the present document gives recommendations for when profile information should be sent.

It is up to the PSS server to retrieve the device capability profiles using the URLs in the "x-wap-profile" header. The PSS server is also responsible to merge the profiles then received. If the "x-wap-profile-diff" header is present it must also merge that information with the retrieved profiles. This functionality is defined in clause 5.2.6.

It should be noted that it is up the implementation of the mobile terminal what URLs to send in the "x-wap-profile" header. For instance, a terminal could just send one URL that points to a complete description of its capabilities. Another terminal might provide one URL that points to a description of the terminal hardware. A second URL that points to a description of a particular software version of the streaming application, and a third URL that points to the description of a hardware or software plug-in that is currently added to the standard configuration of that terminal. From this example it becomes clear that sending URLs from the mobile terminal to the server is good enough not only for static profiles but that it can also handle re-configurations of the mobile terminal such as software version changes, software plug-ins, hardware upgrades, etc.

As described above the list of URLs in the x-wap-profile header is a powerful tool to handle dynamic changes of the mobile terminal. The "x-wap-profile-diff" header could also be used to facilitate the same functionality. To use the "x-wap-profile-diff" header to e.g. send a complete profile (no URL present at all in the "x-wap-profile header") or updates as a result of e.g. a hardware plug-in is not recommended unless some compression scheme is applied over the air-interface. The reason is of course that the size of a profile may be large.

A.4.7 Example of a PSS device capability description

The following is an example of a device capability profile as it could be available from a device profile server. The XML document includes the description of the imaginary "Phone007" phone.

Instead of a single XML document the description could also be spread over several files. The PSS server would need to retrieve these profiles separately in this case and would need to merge them. For instance, this would be useful when device capabilities of this phone that are related to streaming would differ among different versions of the phone. In this case the part of the profile for streaming would be separated from the rest into its own profile document. This separation allows describing the difference in streaming capabilities by providing multiple versions of the profile document for the streaming capabilities.

```
<?xml version="1.0"?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:ccpp="http://www.w3.org/2002/11/08-ccpp-ns#"
         xmlns:prf="http://www.wapforum.org/profiles/UAPROF/ccppschema-20010330#"
         xmlns:pss6="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#">

  <rdf:Description rdf:about="http://www.bar.com/Phones/Phone007">

    <ccpp:component>
      <rdf:Description rdf:ID="HardwarePlatform">
      <rdf:type rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#HardwarePlatform" />
        <prf:BitsPerPixel>4</prf:BitsPerPixel>
        <prf:ColorCapable>Yes</prf:ColorCapable>
        <prf:PixelAspectRatio>1x2</prf:PixelAspectRatio>
        <prf:PointingResolution>Pixel</prf:PointingResolution>

        <prf:Model>Phone007</prf:Model>
        <prf:Vendor>Ericsson</prf:Vendor>
      </rdf:Description>
    </ccpp:component>

    <ccpp:component>
      <rdf:Description rdf:ID="SoftwarePlatform">
      <rdf:type rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#SoftwarePlatform" />
        <prf:CcppAccept-Charset>
          <rdf:Bag>
            <rdf:li>UTF-8</rdf:li>
            <rdf:li>ISO-10646-UCS-2</rdf:li>
          </rdf:Bag>
        </prf:CcppAccept-Charset>
        <prf:CcppAccept-Encoding>
          <rdf:Bag>
            <rdf:li>base64</rdf:li>
            <rdf:li>quoted-printable</rdf:li>
          </rdf:Bag>
        </prf:CcppAccept-Encoding>
```

```
      <prf:CcppAccept-Language>
        <rdf:Seq>
          <rdf:li>en</rdf:li>
          <rdf:li>se</rdf:li>
        </rdf:Seq>
      </prf:CcppAccept-Language>
    </rdf:Description>
  </ccpp:component>

  <ccpp:component>
    <rdf:Description rdf:ID="PssCommon">
    <rdf:type rdf:resource="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#PssCommon" />
      <pss6:AudioChannels>Stereo</pss6:AudioChannels>
      <pss6:MaxPolyphony>24</pss6:MaxPolyphony>
      <pss6:PssVersion>3GPP-R6</pss6:PssVersion>
      <pss6:RenderingScreenSize>160x120</pss6:RenderingScreenSize>
    </rdf:Description>
  </ccpp:component>

  <ccpp:component>
    <rdf:Description rdf:ID="Streaming">
    <rdf:type rdf:resource="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#Streaming" />
      <pss6:ThreeGPPLinkChar>Yes</pss6:ThreeGPPLinkChar>
      <pss6:AdaptationSupport>Yes</pss6:AdaptationSupport>
      <pss6:ExtendedRtcpReports>Yes</pss6:ExtendedRtcpReports>
      <pss6:MediaAlternatives>Yes</pss6:MediaAlternatives>
      <pss6:RtpProfiles>
        <rdf:Bag>
          <rdf:li>RTP/AVP</rdf:li>
          <rdf:li>RTP/AVPF</rdf:li>
        </rdf:Bag>
      </pss6:RtpProfiles>
      <pss6:VideoPreDecoderBufferSize>30720</pss6:VideoPreDecoderBufferSize>
      <pss6:VideoInitialPostDecoderBufferingPeriod>0</pss6:VideoInitialPostDecoderBufferingPeriod>
      <pss6:VideoDecodingByteRate>16000</pss6:VideoDecodingByteRate>
      <pss6:StreamingAccept>
        <rdf:Bag>
          <rdf:li>audio/AMR</rdf:li>
          <rdf:li>audio/AMR-WB;octet-alignment=1</rdf:li>
          <rdf:li>video/H263-2000;profile=0;level=45</rdf:li>
          <rdf:li>video/H263-2000;profile=3;level=45</rdf:li>
          <rdf:li>video/MP4V-ES</rdf:li>
        </rdf:Bag>
      </pss6:StreamingAccept>
    </rdf:Description>
  </ccpp:component>

  <ccpp:component>
    <rdf:Description rdf:ID="ThreeGPFileFormat">
    <rdf:type rdf:resource="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#ThreeGPFileFormat" />
      <pss6:Brands>
        <rdf:Bag>
          <rdf:li>3gp4</rdf:li>
          <rdf:li>3gp5</rdf:li>
          <rdf:li>3gp6</rdf:li>
          <rdf:li>3gr6</rdf:li>
        </rdf:Bag>
      </pss6:Brands>
      <pss6:ThreeGPAccept>
        <rdf:Bag>
          <rdf:li>audio/AMR</rdf:li>
          <rdf:li>audio/AMR-WB;octet-alignment=1</rdf:li>
          <rdf:li>video/H263-2000;profile=0;level=45</rdf:li>
          <rdf:li>video/H263-2000;profile=3;level=45</rdf:li>
          <rdf:li>video/Timed-Text</rdf:li>
        </rdf:Bag>
      </pss6:ThreeGPAccept>
    </rdf:Description>
  </ccpp:component>

  <ccpp:component>
    <rdf:Description rdf:ID="PssSmil">
    <rdf:type rdf:resource="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#PssSmil" />
      <pss6:SmilAccept>
        <rdf:Bag>
          <rdf:li>Streaming-Media</rdf:li>
          <rdf:li>video/3gpp</rdf:li>
          <rdf:li>audio/AMR</rdf:li>
```

```
            <rdf:li>audio/sp-midi</rdf:li>
          </rdf:Bag>
        </pss6:SmilAccept>
        <pss6:SmilAccept-Subset>
          <rdf:Bag>
            <rdf:li>JPEG-PSS</rdf:li>
          </rdf:Bag>
        </pss6:SmilAccept-Subset>
        <pss6:SmilBaseSet>SMIL-3GPP-R6</pss6:SmilBaseSet>
        <pss6:SmilModules>
          <rdf:Bag>
            <rdf:li>BasicTransitions</rdf:li>
            <rdf:li>MulitArcTiming</rdf:li>
          </rdf:Bag>
        </pss6:SmilModules>
      </rdf:Description>
    </ccpp:component>

  </rdf:Description>
</rdf:RDF>
```

Annex B (informative): SMIL authoring guidelines

The SMIL authoring guidelines are given in [52].

Annex C (normative): MIME media types

C.1 (void)

C.2 MIME media type sp-midi

MIME media type name: audio
MIME subtype name: sp-midi

Required parameters: none

Optional parameters: none

NOTE: The above text will be replaced with a reference to the RFC describing the sp-midi MIME media type as soon as this becomes available.

C.3 MIME media type mobile-xmf

MIME media type name: audio
MIME subtype name: mobile-xmf

Required parameters: none

Optional parameters:

prl:
prl is a string (inside double quotation marks ") containing the playback resources included in all Content Description MetaDataItems of the Mobile XMF file. The string contains two digit hexadecimal numbers representing data bytes from the Content Description Meta Data. The same resource is listed only once. A playback resource contains two parts: a prefix and data. If the file includes Playback Resource Lists such as [00h 01h 00h 02h] and [00h 01h 00h 03h], the corresponding prl is '000100020003' containing playback resources 01, 02, and 03 with the prefix 00.

minimum-pr:
minimum-pr is a string containing the Maximum Instantaneous Resource (MIR) values from the first row of all MIR Count Tables corresponding to the playback resources listed in prl. Only the largest value from the values of the same resource is chosen. If the file includes first rows of MIR Count Tables such as [02h 00h] and [01h 01h] corresponding to the above Playback Resource Lists, the corresponding minimum-pr is '020001'. (02 is the largest of 2 and 1, 00 is the largest of 0, and 01 is the largest of 1.) minimum-pr requires the use of prl and the values in minimum-pr must be in the same order as the resources in prl. minimum-pr is the most important of minimum-pr and total-pr, because it defines the minimum playback requirements.

total-pr:
total-pr is a string containing the MIR values from the last row of all MIR Count Tables corresponding to the playback resources listed in prl. Only the largest value from the values of the same resource is chosen. If the file includes last rows of MIR Count Tables such as [05h 02h] and [06h 01h] corresponding to the above Playback Resource Lists, the corresponding total-pr is '060201'. (06 is the largest of 5 and 6, 02 is the largest of 2, and 01 is the largest of 1.) total-pr requires the use of prl and the values in total-pr must be in the same order as the resources in prl.

NOTE: The above text will be replaced with a reference to the RFC describing the mobile-xmf MIME media type as soon as this becomes available.

C.4 (void)

Annex D (normative): 3GP files – codecs and identification

The definition of the 3GPP file format, including codec registration and file identification, is given in [50]. The timed text format is defined in [51].

Annex E (normative): RTP payload format and file storage format for AMR and AMR-WB audio

The AMR and AMR-WB speech codec RTP payload, storage format and MIME type registration are specified in [11].

Annex F (normative): RDF schema for the PSS base vocabulary

```
<?xml version="1.0"?>

<!--
     This document is the RDF Schema for Packet-switched Streaming
     Service (PSS)-specific vocabulary as defined in 3GPP TS 26.234
     Release 7 (in the following "the specification").

     The URI for unique identification of this RDF Schema is
       http://www.3gpp.org/profiles/PSS/ccppschema-PSS7#

     This RDF Schema includes the same information as the respective
     chapter of the specification. Greatest care has been taken to keep
     the two documents consistence. However, in case of any divergence
     the specification takes presidence.

     All reference in this RDF Schmea are to be interpreted relative to
     the specification. This means all references using the form
     [ref] are defined in chapter 2 "References" of the specification.
     All other references refer to parts within that document.

     Note: This Schemas has been aligned in structure and base
     vocabulary to the RDF Schema used by UAProf [40].

-->

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" >

<!-- ******************************************************************* -->
<!-- ***** Properties shared among the components***** -->

  <rdf:Description rdf:ID="defaults">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:domain rdf:resource="#ThreeGPFileFormat"/>
    <rdfs:domain rdf:resource="#PssSmil"/>
    <rdfs:comment>
      An attribute used to identify the default capabilities.
    </rdfs:comment>
  </rdf:Description>

<!-- ******************************************************************* -->
<!-- ***** Component Definitions ***** -->

  <rdf:Description rdf:ID="PssCommon">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#Component"/>
    <rdfs:label>Component: PssCommon</rdfs:label>
    <rdfs:comment>
      The PssCommon component specifies the base vocabulary common for all
      PSS applications, in contrast to application-specific parts of the PSS
      base vocabulary which are described by the Streaming, ThreeGPFileFormat and
      PssSmil components defined below.

      PSS servers supporting capability exchange should understand the attributes
      in this component as explained in detail in 3GPP TS 26.234 Release 7..
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="Streaming">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#Component"/>
    <rdfs:label>Component: Streaming</rdfs:label>
    <rdfs:comment>
      The Streaming component specifies the base vocabulary for pure RTSP/RTP-
      based streaming in PSS.
```

```
      PSS servers supporting capability exchange should understand the attributes
      in this component as explained in detail in 3GPP TS 26.234 Release 7.
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="ThreeGPFileFormat">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#Component"/>
    <rdfs:label>Component: ThreeGPFileFormat</rdfs:label>
    <rdfs:comment>
      The ThreeGPFileFormat component specifies the base vocabulary for 3GP file
      download or progressive download in PSS.

      PSS servers supporting capability exchange should understand the attributes
      in this component as explained in detail in 3GPP TS 26.234 Release 7.
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="PssSmil">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#Component"/>
    <rdfs:label>Component: PssSmil</rdfs:label>
    <rdfs:comment>
      The PssSmil component specifies the base vocabulary for SMIL presentations
      in PSS. Note that capabibilites regarding streaming and 3GP files that are
      part of a SMIL presentation are expressed by the vocabularies specified by
      the Streaming and ThreeGPFileFormat components, respectively.

      PSS servers supporting capability exchange should understand the attributes
      in this component as explained in detail in 3GPP TS 26.234 Release 7.
    </rdfs:comment>
  </rdf:Description>

<!-- **
     ** In the following property definitions, the defined types
     ** are as follows:
     **
     ** Number: A positive integer
     ** [0-9]+
     ** Boolean: A yes or no value
     ** Yes|No
     ** Literal: An alphanumeric string
     ** [A-Za-z0-9/.\-_]+
     ** Dimension: A pair of numbers
     ** [0-9]+x[0-9]+
     **
-->

<!-- ******************************************************************* -->
<!-- ***** Component: PssCommon ***** -->

  <rdf:Description rdf:ID="AudioChannels">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: This attribute describes the stereophonic capability of the
      natural audio device. The only legal values are "Mono" and "Stereo".

      Type: Literal
      Resolution: Locked
      Examples: "Mono", "Stereo"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="MaxPolyphony">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: The MaxPolyphony attribute refers to the maximal polyphony
      that the synthetic audio device supports as defined in [44]. Legal values
      are integer between 5 to 24.
      NOTE: MaxPolyphony attribute can be used to signal the maximum polyphony
            capabilities supported by the PSS client. This is a complementary
            mechanism for the delivery of compatible SP-MIDI content and thus
            the PSS client is required to support Scalable Polyphony MIDI i.e.
            Channel Masking defined in [44].
```

```
      Type: Number
      Resolution: Locked
      Examples: 8
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="NumOfGM1Voices">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: The NumOfGM1Voices attribute refers to the maximum number
      of simultaneous GM1   voices that the synthetic audio engine supports.
      Legal values are integers greater or equal than 5.

      Type: Number
      Resolution: Locked
      Examples: 24
    </rdfs:comment>
  </rdf:Description>

<rdf:Description rdf:ID="NumOfMobileDLSVoicesWithoutOptionalBlocks">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: The NumOfMobileDLSVoicesWithoutOptionalBlocks attribute
      refers to the maximum number of simultaneous voices without optional
      group of processing blocks that the synthetic audio engine supports.
      Legal values are integers greater or equal than 5.

      Type: Number
      Resolution: Locked
      Examples: 24
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="NumOfMobileDLSVoicesWithOptionalBlocks">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: The NumOfMobileDLSVoicesWithOptionalBlocks attribute refers
      to the maximum number of simultaneous voices with optional group of
      processing blocks that the synthetic audio engine supports. This attribute
      is set to zero for devices that do not support the optional group of
      processing blocks. Legal values are integers greater or equal than 0.

      Type: Number
      Resolution: Locked
      Examples: 24
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="PssVersion">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: Latest PSS version supported by the client. Legal
      values are "3GPP-R4", "3GPP-R5", "3GPP-R6", "3GPP-R7" and so forth.

      Type: Literal
      Resolution: Locked
      Examples: "3GPP-R5", "3GPP-R6"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="RenderingScreenSize">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: The rendering size of the device's screen in unit of
      pixels available for PSS media presentation. The horizontal size is
      given followed by the vertical size. Legal values are pairs of integer
      values equal or greater than zero. A value equal "0x0"means that there
      exists no display or just textual output is supported.

      Type: Dimension
      Resolution: Locked
      Examples: "160x120"
```

```
    </rdfs:comment>
  </rdf:Description>

<!-- ************************************************************** -->
<!-- ***** Component: Streaming ***** -->

  <rdf:Description rdf:ID="StreamingAccept">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: List of content types (MIME types) relevant for streaming
      over RTP supported by the PSS application. Content types listed shall be
      possible to stream over RTP. For each content type a set of MIME parameters
      can be specified to signal receiver capabilities. A content type that
      supports multiple parameter sets may occur several times in the list.
      Legal values are lists of MIME types with related parameters.

      Type: Literal (bag)
      Resolution: Append
      Examples: "audio/AMR-WB;octet-alignment=1,application/smil"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="StreamingAccept-Subset">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: List of content types for which the PSS application supports
      a subset. MIME types can in most cases effectively be used to express
      variations in support for different media types. Many MIME types, e.g.
      AMR-WB has several parameters that can be used for this purpose. There
      may exist content types for which the PSS application only supports a
      subset and this subset cannot be expressed with MIME-type parameters.
      In these cases the attribute StreamingAccept-Subset is used to describe
      support for a subset of a specific content type. If a subset of a specific
      content type is declared in StreamingAccept-Subset, this means that
      StreamingAccept-Subset has precedence over StreamingAccept.
      StreamingAccept shall always include the corresponding content types for
      which StreamingAccept-Subset specifies subsets of.
      No legal values are currently defined.

      Type: Literal (bag)
      Resolution: Locked
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="LinkChar">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device supports the
      3GPP-Link-Char header according to clause 10.2.1.1 of the specification.
      Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Override
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="AdaptationSupport">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device supports
      client buffer feedback signaling according to clause 10.2.3 of the
      specification. Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Locked
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="QoESupport">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
```

```
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device supports
      QoE signaling according to clauses 5.3.2.3, 5.3.3.6, and 11 of the
      specification. Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Locked
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="ExtendedRtcpReports">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device supports
      extended RTCP reports according to clause 6.2.3.1 of the specification.
      Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Locked
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="RtpRetransmission">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device supports RTP
      retransmission according to clause 6.2.3.3 of the specification.
      Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Locked
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="MediaAlternatives">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device interprets the
      SDP attributes "alt", "alt-default-id", and "alt-group", defined in
      clauses 5.3.3.3 and 5.3.3.4 of the specification.
      Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Override
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="RtpProfiles">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute lists the supported RTP profiles. Legal
      values are profile names registered through the Internet Assigned Numbers
      Authority (IANA), www.iana.org.

      Type: Literal (bag)
      Resolution: Append
      Examples: "RTP/AVP,RTP/AVPF"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="StreamingOmaDrm">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: Indicates whether the device supports streamed OMA DRM
      protected content, as defined by OMA and Annex K. Legal values are OMA
      Version numbers supported as a floating number. 0.0 indicates no support.
```

```
    Type: Literal (bag)
    Resolution: Locked
    Examples: "2.0"
  </rdfs:comment>
</rdf:Description>

<rdf:Description rdf:ID="PSSIntegrity">
  <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
  <rdfs:domain rdf:resource="#Streaming"/>
  <rdfs:comment>
    Description: Indicates whether the device supports integrity protection
    for streamed content as defined by Annex K.2. Legal values are "Yes" and
    "No".

    Type: Literal
    Resolution: Locked
    Examples: "Yes"
  </rdfs:comment>
</rdf:Description>

<rdf:Description rdf:ID="VideoDecodingByteRate">
  <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:domain rdf:resource="#Streaming"/>
  <rdfs:comment>
    Description: If Annex G is not supported, the attribute has no meaning.
    If Annex G is supported, this attribute defines the peak decoding byte
    rate the PSS client is able to support. In other words, the PSS client
    fulfils the requirements given in Annex G with the signalled peak decoding
    byte rate. The values are given in bytes per second and shall be greater
    than or equal to 16000. According to Annex G, 16000 is the default peak
    decoding byte rate for the mandatory video codec profile and level
    (H.263 Profile 0 Level 45). Legal values are integer values greater than
    or equal to 16000.

    Type: Number
    Resolution: Locked
    Examples: "16000"
  </rdfs:comment>
</rdf:Description>

<rdf:Description rdf:ID="VideoInitialPostDecoderBufferingPeriod">
  <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:domain rdf:resource="#Streaming"/>
  <rdfs:comment>
    Description: If Annex G is not supported, the attribute has no
    meaning. If Annex G is supported, this attribute defines the
    maximum initial post-decoder buffering period of video. Values are
    interpreted as clock ticks of a 90-kHz clock. In other words, the
    value is incremented by one for each 1/90 000 seconds. For
    example, the value 9000 corresponds to 1/10 of a second initial
    post-decodder buffering. Legal values are all integer values equal
    to or greater than zero.

    Type: Number
    Resolution: Locked
    Examples: "9000"
  </rdfs:comment>
</rdf:Description>

<rdf:Description rdf:ID="VideoPreDecoderBufferSize">
  <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:domain rdf:resource="#Streaming"/>
  <rdfs:comment>
    Description: This attribute signals if the optional video
    buffering requirements defined in Annex G are supported. It also
    defines the size of the hypothetical pre-decoder buffer defined in
    Annex G. A value equal to zero means that Annex G is not
    supported. A value equal to one means that Annex G is
    supported. In this case the size of the buffer is the default size
    defined in Annex G.  A value equal to or greater than the default
    buffer size defined in Annex G means that Annex G is supported and
    sets the buffer size to the given number of octets. Legal values are all
    integer values equal to or greater than zero. Values greater than
    one but less than the default buffer size defined in Annex G are
    not allowed.
```

```
      Type: Number
      Resolution: Locked
      Examples: "0", "4096"
    </rdfs:comment>
  </rdf:Description>

<!-- ********************************************************************** -->
<!-- ***** Component: ThreeGPFileFormat ***** -->

  <rdf:Description rdf:ID="Brands">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#ThreeGPFileFormat"/>
    <rdfs:comment>
      Description: This attribute lists the supported 3GP profiles identified
      by brand. Legal values are brand identifiers according to 5.3.4 and 5.4
      in [50].

      Type: Literal (bag)
      Resolution: Append
      Examples: "3gp4,3gp5,3gp6,3gr6,3gp7,3gr7,3ge7"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="ThreeGPAccept">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#ThreeGPFileFormat"/>
    <rdfs:comment>
      Description: List of content types (MIME types) that can be included
      in a 3GP file and handled by the PSS application. If the identifier
      "Streaming-Media" is included, streaming media can be included in the
      presentation, e.g. in DIMS. Details on the streaming support can then be
      found in the Streaming component. For each content
      type a set of supported parameters can be given. A content type that
      supports multiple parameter sets may occur several times in the list.

      Type: Literal (bag)
      Resolution: Append
      Examples: "video/H263-2000;profile=0;level=45,audio/AMR"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="ThreeGPAccept-Subset">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#ThreeGPFileFormat"/>
    <rdfs:comment>
      Description: List of content types for which the PSS application
      supports a subset. MIME types can in most cases effectively be used
      to express variations in support for different media types. Many MIME
      types have several parameters that can be used for this purpose. There
      may exist content types for which the PSS application only supports a
      subset and this subset cannot be expressed with MIME type parameters.
      In these cases the attribute ThreeGPAccept-Subset is used to describe
      support for a subset of a specific content type. If a subset of a
      specific content type is declared in ThreeGPAccept-Subset, this means that
      ThreeGPAccept-Subset has precedence over ThreeGPAccept. ThreeGPAccept shall always
      include the corresponding content types for which ThreeGPAccept-Subset
      specifies subsets of. No legal values are currently defined.

      Type: Literal (bag)
      Resolution: Locked
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="ThreeGPOmaDrm">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#ThreeGPFileFormat"/>
    <rdfs:comment>
      Description: List of the OMA DRM versions that is supported to be used
      for DRM protection of content present in the 3GP file format. Legal values
      are OMA DRM version numbers as floating values. 0.0 indicates no support.

      Type: Literal (bag)
      Resolution: Locked
      Examples: "2.0"
```

```
    </rdfs:comment>
  </rdf:Description>

<!-- ********************************************************************** -->
<!-- ***** Component: PssSmil ***** -->

  <rdf:Description rdf:ID="SmilAccept">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#PssSmil"/>
    <rdfs:comment>
      Description: List of content types (MIME types) that can be part of a
      SMIL presentation. The content types included in this attribute can be
      rendered in a SMIL presentation. If video/3gpp (or audio/3gpp) is
      included, downloaded 3GP files can be included in a SMIL presentation.
      Details on the 3GP file support can then be found in the ThreeGPFileFormat
      component. If the identifier "Streaming-Media" is included, streaming
      media can be included in the SMIL presentation. Details on the
      streaming support can then be found in the Streaming component.
      For each content type a set of supported parameters can be given.
      A content type that supports multiple parameter sets may occur several
      times in the list. Legal values are lists of MIME types with related
      parameters and the "Streaming-Media" identifier.

      Type: Literal (bag)
      Resolution: Append
      Examples: "image/gif,image/jpeg,Streaming-Media"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="SmilAccept-Subset">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#PssSmil"/>
    <rdfs:comment>
      Description: List of content types for which the PSS application
      supports a subset. MIME types can in most cases effectively be used to
      express variations in support for different media types. Many MIME types
      have several parameters that can be used for this purpose. There may
      exist content types for which the PSS application only supports a subset
      and this subset cannot be expressed with MIME-type parameters. In these
      cases the attribute SmilAccept-Subset is used to describe support for a
      subset of a specific content type. If a subset of a specific content type
      is declared in SmilAccept-Subset, this means that SmilAccept-Subset has
      precedence over SmilAccept. SmilAccept shall always include the
      corresponding content types for which SmilAccept-Subset specifies subsets
      of.

      The following values are defined:
        - "JPEG-PSS": Only the two JPEG modes described in clause 7.5 of the
                      specifictaion are supported.
        - "SVG-Tiny"
        - "SVG-Basic"

      Subset identifiers and corresponding semantics shall only be defined by
      the TSG responsible for the present document.

      Type: Literal (bag)
      Resolution: Append
      Examples: "JPEG-PSS,SVG-Tiny"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="SmilBaseSet">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssSmil"/>
    <rdfs:comment>
      Description: Indicates a base set of SMIL 2.0 modules that the client
      supports. Leagal values are the following pre-defined identifiers:
      "SMIL-3GPP-R4" and "SMIL-3GPP-R5" indicate all SMIL 2.0 modules required
      for SMIL scene-description support according to clause 8 of Release 4 and
      Release 5, respectively, of TS 26.234. "SMIL-3GPP-R6" and "SMIL-3GPP-R7"
      indicate all SMIL 2.0 modules required for SMIL scene description support
      according to Release 6 and Release 7, respectively, of clause 8 of the
      specification and of TS 26.246 [52].

      Type: Literal
      Resolution: Locked
```

```
      Examples: "SMIL-3GPP-R4", "SMIL-3GPP-R5"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="SmilModules">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#PssSmil"/>
    <rdfs:comment>
      Description: This attribute defines a list of SMIL 2.0 modules
      supported by the client. If the SmilBaseSet is used those modules
      do not need to be explicitly listed here. In that case only
      additional module support needs to be listed. Legal values are all
      SMIL 2.0 module names defined in the SMIL 2.0 recommendation [31],
      section 2.3.3, table 2.

      Type: Literal (bag)
      Resolution: Locked
      Examples: "BasicTransitions,MulitArcTiming"
    </rdfs:comment>
  </rdf:Description>

</rdf:RDF>
```

Annex G (normative): Buffering of video

G.1 Introduction

This annex describes video buffering requirements in the PSS. As defined in clause 7.4 of the present document, support for the annex is optional and may be signalled in the PSS capability exchange and in the SDP. This is described in clause 5.2 and clause 5.3.3 of the present document. When the annex is in use, the content of the annex is normative. In other words, PSS clients shall be capable of receiving an RTP packet stream that complies with the specified buffering model and PSS servers shall verify that the transmitted RTP packet stream complies with the specified buffering model.

G.2 PSS Buffering Parameters

The behaviour of the PSS buffering model is controlled with the following parameters: the initial pre-decoder buffering period, the initial post-decoder buffering period, the size of the hypothetical pre-decoder buffer, the peak decoding byte rate, and the decoding macroblock rate. The default values of the parameters are defined below.

- The default initial pre-decoder buffering period is 1 second.
- The default initial post-decoder buffering period is zero.
- The default size of the hypothetical pre-decoder buffer is defined according to the maximum video bit-rate according to the table below:

Table G.1: Default size of the hypothetical pre-decoder buffer

Maximum video bit-rate	Default size of the hypothetical pre-decoder buffer
65536 bits per second	20480 bytes
131072 bits per second	40960 bytes
Undefined	51200 bytes

- The maximum video bit-rate can be signalled in the media-level bandwidth attribute of SDP as defined in clause 5.3.3 of this document. If the video-level bandwidth attribute was not present in the presentation description, the maximum video bit-rate is defined according to the video coding profile and level in use.
- The size of the hypothetical post-decoder buffer is an implementation-specific issue. The buffer size can be estimated from the maximum output data rate of the decoders in use and from the initial post-decoder buffering period.
- By default, the peak decoding byte rate is defined according to the video coding profile and level in use. For example, H.263 Level 45 requires support for bit-rates up to 128000 bits per second. Thus, the peak decoding byte rate equals to 16000 bytes per second.
- The default decoding macroblock rate is defined according to the video coding profile and level in use. If MPEG-4 Visual is in use, the default macroblock rate equals to VCV decoder rate. If H.263 is in use, the default macroblock rate equals to (1 / minimum picture interval) multiplied by number of macroblocks in maximum picture format. For example, H.263 Profile 0 Level 45 requires support for picture formats up to QCIF and minimum picture interval down to 2002 / 30000 sec. Thus, the default macroblock rate would be 30000 x 99 / 2002 ≈ 1484 macroblocks per second.

PSS clients may signal their capability of providing larger buffers and faster peak decoding byte rates in the capability exchange process described in clause 5.2 of the present document. The average coded video bit-rate should be smaller than or equal to the bit-rate indicated by the video coding profile and level in use, even if a faster peak decoding byte rate were signalled.

Initial parameter values for each stream can be signalled within the SDP description of the stream. Signalled parameter values override the corresponding default parameter values. The values signalled within the SDP description guarantee pauseless playback from the beginning of the stream until the end of the stream (assuming a constant-delay reliable transmission channel).

PSS servers may update parameter values in the response for an RTSP PLAY request. If an updated parameter value is present, it shall replace the value signalled in the SDP description or the default parameter value in the operation of the PSS buffering model. An updated parameter value is valid only in the indicated playback range, and it has no effect after that. Assuming a constant-delay reliable transmission channel, the updated parameter values guarantee pauseless playback of the actual range indicated in the response for the PLAY request. The indicated pre-decoder buffer size and initial post-decoder buffering period shall be smaller than or equal to the corresponding values in the SDP description or the corresponding default values, whichever ones are valid. The header fields for RTSP are specified in clause 5.3.2.4.

The following example plays the whole presentation starting at SMPTE time code 0:10:20 until the end of the clip. The playback is to start at 15:36 on 23 Jan 1997. The suggested initial pre-decoder buffering period is half a second.

```
C->S: PLAY rtsp://audio.example.com/twister.en RTSP/1.0
      CSeq: 833
      Session: 12345678
      Range: smpte=0:10:20-;time=19970123T153600Z
      User-Agent: TheStreamClient/1.1b2

S->C: RTSP/1.0 200 OK
      CSeq: 833
      Date: 23 Jan 1997 15:35:06 GMT
      Range: smpte=0:10:22-;time=19970123T153600Z
      x-initpredecbufperiod: 45000
```

G.3 PSS server buffering verifier

The PSS server buffering verifier is specified according to the PSS buffering model. The model is based on two buffers and two timers. The buffers are called the hypothetical pre-decoder buffer and the hypothetical post-decoder buffer. The timers are named the decoding timer and the playback timer.

The PSS buffering model is presented below.

1. The buffers are initially empty.
2. A PSS Server adds each transmitted RTP packet having video payload to the pre-decoder buffer immediately when it is transmitted. All protocol headers at RTP or any lower layer are removed.
3. Data is not removed from the pre-decoder buffer during a period called the initial pre-decoder buffering period. The period starts when the first RTP packet is added to the buffer.
4. When the initial pre-decoder buffering period has expired, the decoding timer is started from a position indicated in the previous RTSP PLAY request.
5. Removal of a video frame is started when both of the following two conditions are met: First, the decoding timer has reached the scheduled playback time of the frame. Second, the previous video frame has been totally removed from the pre-decoder buffer.
6. The duration of frame removal is the larger one of the two candidates: The first candidate is equal to the number of macroblocks in the frame divided by the decoding macroblock rate. The second candidate is equal to the number of bytes in the frame divided by the peak decoding byte rate. When the coded video frame has been removed from the pre-decoder buffer entirely, the corresponding uncompressed video frame is located into the post-decoder buffer.
7. Data is not removed from the post-decoder buffer during a period called the initial post-decoder buffering period. The period starts when the first frame has been placed into the post-decoder buffer.
8. When the initial post-decoder buffering period has expired, the playback timer is started from the position indicated in the previous RTSP PLAY request.
9. A frame is removed from the post-decoder buffer immediately when the playback timer reaches the scheduled playback time of the frame.

10. Each RTSP PLAY request resets the PSS buffering model to its initial state.

A PSS server shall verify that a transmitted RTP packet stream complies with the following requirements:

- The PSS buffering model shall be used with the default or signalled buffering parameter values. Signalled parameter values override the corresponding default parameter values.
- The occupancy of the hypothetical pre-decoder buffer shall not exceed the default or signalled buffer size.
- Each frame shall be inserted into the hypothetical post-decoder buffer before or on its scheduled playback time.

G.4 PSS client buffering requirements

When the annex is in use, the PSS client shall be capable of receiving an RTP packet stream that complies with the PSS server buffering verifier when the RTP packet stream is carried over a constant-delay reliable transmission channel. Furthermore, the video decoder of the PSS client, which may include handling of post-decoder buffering, shall output frames at the correct rate defined by the RTP time-stamps of the received packet stream.

Annex H (informative): Content creator guidelines for the synthetic audio medium type

It is recommended that the first element of the MIP (Maximum Instantaneous Polyphony) message of the SP-MIDI content intended for synthetic audio PSS/MMS should be no more than 5. For instance the following MIP figures {4, 9, 10, 12, 12, 16, 17, 20, 26, 26, 26} complies with the recommendation whereas {6, 9, 10, 12, 12, 16, 17, 20, 26, 26, 26} does not.

Annex I (informative): Void

Annex J (informative): Mapping of SDP parameters to UMTS QoS parameters

This Annex gives recommendation for the mapping rules needed by the PSS applications to request the appropriate QoS from the UMTS network (see Table J.1).

Table J.1: Mapping of SDP parameters to UMTS QoS parameters for PSS

QoS parameter	Parameter value	comment
Delivery of erroneous SDUs	'No'	
Delivery order	'No'	
Traffic class	"Streaming class"	
Maximum SDU size	1400 bytes	According to RFC 2460 the SDU size must not exceed 1500 octets. A packet size of 1400 guarantees efficient transportation.
Guaranteed bit rate for downlink	1.025 * session bandwidth	This session bandwidth is calculated from the SDP media level bandwidth values.
Maximum bit rate for downlink	Equal or higher to guaranteed bit rate in downlink	
Guaranteed bit rate for uplink	0.025 * session bandwidth	
Maximum bit rate for uplink	Equal or higher to guaranteed bit rate in uplink	
Residual BER	1*10-5	16 bit CRC should be enough
SDU error ratio	1*10-4 or better	
Traffic handling priority	Subscribed traffic handling priority	Ignored
Transfer delay	2 sec.	

Annex K (normative): Void (Substituted by Annex R)

Annex L (informative): SVG Tiny 1.2 content creation guidelines

L.1 Feature analysis

This clause provides an analysis of SVG Tiny 1.2 features in Table L.1.

Table L.1: Feature analysis of SVG Tiny 1.2

Element / feature	Status	Comment
animate*	Should be used with caution.	In conjunction with other expensive features, which are OK for static scenes, animation may yield scenes with insufficient performance.
Animation	To be used sparingly.	High potential for performance hit, should be used with caution.
audio	No constraint	
desc / title / metadata	No constraint	
Text Area	Should not be animated continuously.	
SVG fonts	Should be used sparingly.	Use of SVG fonts may lead to poor readability, in which case device font support should be preferred. Also, SVG fonts increase the size of content and thus download times.
foreignObject	No constraint	May be safely ignored by SVG Tiny 1.2 implementations as there is no conformant rendering behaviour for foreignObject in SVG Tiny.
a / g / defs	No constraint	
image	Animation of scale and rotation should be used sparingly.	Animation of scale and rotation of an image has a similar CPU requirement than transformed video rendering, and as such should be avoided on most devices.
linearGradient / radialGradient / stop	Should be used sparingly.	This may lead to a significant performance hit on lower/middle-end devices.
complex stroking and transparency	Should be used sparingly.	The screen surface used by transparent objects should be small. Full screen fade-in, fade-out (by simply animating fill-opacity and opacity on images) or cross-fade should be avoided on most devices. Use of complex stroking will incur a significant performance hit.
<all shapes>	No constraint	
prefetch	No constraint	
script	No constraint	
set	No constraint	
solidColor	No constraint	
svg	No constraint	
switch	No constraint	
text / tspan	No constraint	
use	(=embedded image support). To be used sparingly.	High potential for performance hit, should be used with caution.
video	Video could be used with media-handling="pinned" Scaling = 1,1 / Rotation (none/0°) SVG could provide a handling=media rotation for translation, rotation by 90°multiples or scaling. In any case no animation of the transformation.	On devices without hardware acceleration, frame-by-frame scaling, rotation and re-sampling of video is not achievable. Overlaying graphics on video content is considered a very expensive operation and may have significant consequences, such as lack of synchronization and degraded output quality. The video rendering features are highly dependent on the host capabilities and one may expect potential differences between implementations. Content creators should be very cautious when dealing with such functionality.
Inheritance	should use property inheritance with caution	. Authors should be aware that inheritance incurs an execution cost at each frame for the objects in the part of the tree where inheritance is used.

L.2 Recommendations

L.2.1 General

This clause provides detailed recommendations for the usage of SVG Tiny 1.2 features.

L.2.2 Video element

L.2.2.1 Inclusion of the video element in SVG content

The video element should be included within a "switch" element. The feature string for video could be

1. http://www.w3.org/TR/SVG12/feature#3GPPTransformedVideo
2. the feature string for video is http://www.w3.org/TR/SVG12/feature#3GPPVideo
3. or the alternate representation of a "video" element could be an image.

EXAMPLE:

```
<g transform="translate(10,0);scale(1.5)">
  <switch>
    <video
      xlink:href="video.3gp"
      type="video/H263-2000"
      requiredFeatures="http://www.w3.org/TR/SVG12/feature#TransformedVideo"/>
    <video
      xlink:href="video.3gp"
      type="video/H263-2000"
      requiredFeatures="http://www.w3.org/TR/SVG12/feature#Video
      transformBehavior ='pinned"/>
    <image xlink:ref="image.jpg" width="176" height="144">
  </switch>
</g>
```

The above example shows a transformed video. If the PSS client supports "TransformedVideo", the video shall be transformed, if not, a video-enabled PSS client shall display the video without scaling and rotation ("pinned"). Finally, an image shall be displayed if neither one of the above cases is possible at the PSS client.

L.2.2.2 Transformation of video

SVG Tiny 1.2 supports the video element and proper rendering requires video to be subject to transformation just like any other graphics object. This implies that any arbitrary transform can be applied to embedded video content. Dynamic transformation of video content is an expensive operation and therefore would largely (and negatively) impact the frame rate of animated SVG content. This feature is also known to be very complex to be supported among most of the current mobile devices.

SVG Tiny 1.2 does not require transformed video. As a consequence transform video is optional. When optionally applied to video elements, the following transformations and the animations thereof are applicable in increasing complexity order:

1. Translation of the video element shall be applied.
2. Rotation of video by 90°/-90° degrees is permitted.
3. Scaling of the video element is permitted.

NOTE: PSS clients may decide not to apply scaling through the transformBehavior attribute.

Dynamic transformation of video content should be avoided. Overlaying graphics on video content is considered a very expensive operation and may have significant consequences, such as lack of synchronization and degraded output quality and performance.

The video rendering features are highly dependent on the host capabilities and one may expect potential differences between implementations. Content creators should be very cautious when dealing with such functionality.

L.2.3 Animation Element

SVGT1.2 introduces the SMIL animation element, which allows reference to a scene inside of another scene, and to control the time flow of the referred scene as can be done on a video or audio stream.

This feature requires maintaining multiple DOM trees between the referenced and the root or main SVG image. It can potentially lead to memory and performance issues with additional requirements, such as extra data validation/parsing and maintaining multiple buffers/contexts

Recommendation: Content creators should be cautious when using this feature due to the potential negative performance impact.

L.2.4 Void

L.2.5 Transparency, stroking and gradients

SVG Tiny 1.2 supports fill-opacity and stroke-opacity, complex stroking and gradients. Using transparency gradients and complex stroking is known to slow down the rendering in software implementations. Animation should be confined to a small part of the screen as the performance penalty is usually proportional to the surface of the screen impacted by the animation.

For example, animating the size of a gradient may cause excessive memory consumption and performance drop: this happens when the gradient has gradientUnits='objectBoundingBox' and the size of the object is animated, or in other cases when the viewBox of the svg element is animated.

Recommendation: Content creators should be cautious when using these features due to the potential negative performance in software implementations impact by restricting their use to small surfaces and/or refraining from animating them.

L.2.6 Events

SVG Tiny 1.2 supports the following events: mousemove, mouseover, mouseout, mousedown, mouseup, click, DOMActivate, DOMFocusIn, DOMFocusOut, SVGLoad, SVGScroll, SVGResize, SVGZoom, beginEvent, endEvent, repeat, Text events.

Recommendation: Content creators should be aware that some events are not universally available on all platforms, and consequently they should not rely on the use of the following events: mousemove, mouseover, mouseout, mousedown, mouseup, click.

L.2.7 Text Area

SVG Tiny 1.2 enables a block of text and graphics to be rendered inside a single textArea of rectangle shape, while automatically wrapping the objects into lines, using the flowRoot element.

Recommendation: Content creators should be cautious when using this feature due to the potential negative performance impact and refrain from continuous animation of the texArea attributes.

L.2.8 SVG fonts

SVG Tiny 1.2 supports the definition and use of SVG fonts for rendering text. The lack of hinting in SVG fonts means that small text which is antialiased will become unreadable in most cases. This problem is even more evident when text is rotated or animated. Also, SVG fonts increase the size of content and thus download times. In addition, device-native fonts are often a lot faster.

Recommendation: Usage of device or system fonts is recommended. SVG fonts should be used with care.

L.2.9 Bitmap fonts

When using bitmapped fonts to display text, the content author needs to be aware of the limitations. Rotated text using a bitmapped font may be unreadable.

Recommendation: When using bitmapped fonts, content creators should avoid the display of text rotated at an arbitrary angle. Instead, only multiples of 90 degrees should be used to ensure readability.

L.2.10 Animation

SVG animation has a non-uniform frame rate. The overall complexity of a scene determines the animation frame rate. Complex paths, stroking and property inheritance all have a potential negative impact on the complexity of a scene.

Animation of scale and/or rotation of images also have a significant impact on the fluidity of the rendering, as it is very similar to transformed video rendering in CPU requirement (frame-by-frame resizing, rotation and re-sampling of the bitmap).

Recommendation: Content creators should be cautious when designing animated content with lengthy or complex paths, extensive stroking or excessive property inheritance. Content creators should refrain from animating the scale or rotation of images on devices that do not support transformed video.

L.2.11 User interaction and content navigation

Mobile devices do not provide the same amount of screen area and user input means as a PC. When designing interactive content for mobile devices it is therefore important to remember the potential limitations of the target hardware. For example, most mobile phones do not have a pointing device so having small "hot-spots" of user interaction on the screen is not recommended. Also, the user is typically involved in another activity when using a mobile device, unlike a PC where the machine usually has the user's undivided attention.

Recommendation: Content creators need to be aware of any potential limitations and design user interaction and content navigation accordingly.

L.2.12 Inheritance

SVGT1.2 supports a number of properties originally coming from CSS, and these properties (See SVG Tiny 1.2 Appendix L) can be inherited by children of the elements which define them. Authors should be aware that inheritance incurs an execution cost at each frame for the objects in the part of the tree where inheritance is used.

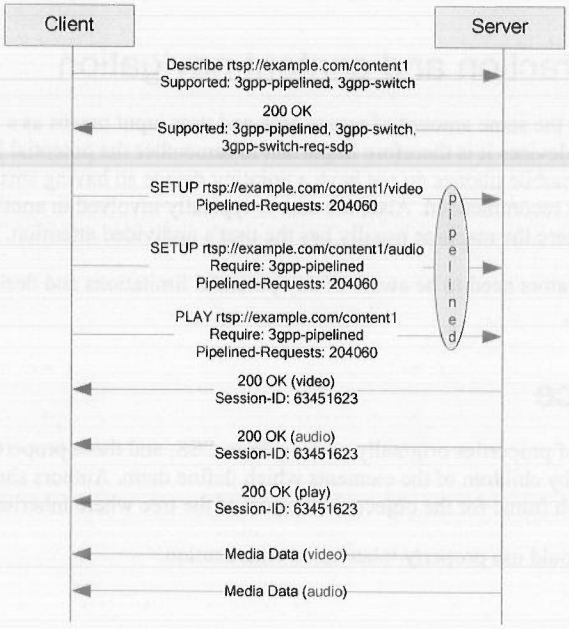
Recommendation: Authors should use property inheritance with caution.

Annex M (informative): Examples for Fast Content Switching and Start-up

M.1 Pipelined Start-up Examples

M.1.1 Successful Pipelined Start-up

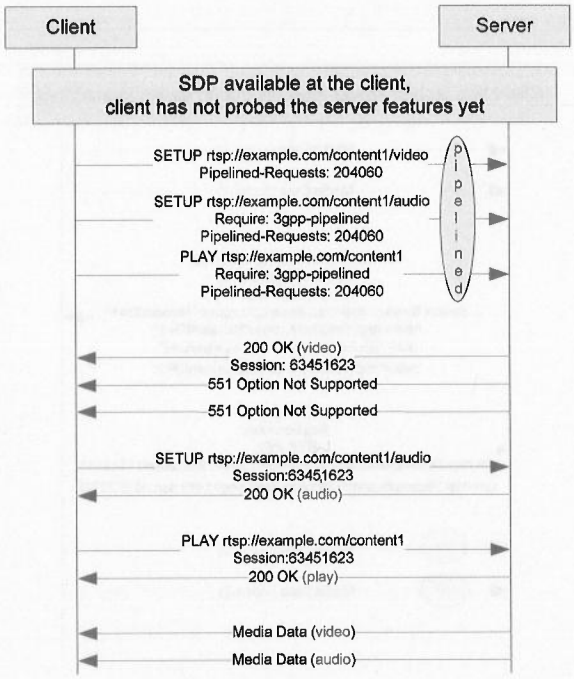
The following is an example depiction of pipelined start-up. The client probes the server features during the optional DECRIBE interaction. Note that the first set-up message does not contain a "Require" header.



M.1.2 Unsuccessful Pipelined Start-up

In this example the client uses the pipelined start-up feature towards a server which does not support this feature. In principle the client may keep knowledge about feature capabilities.

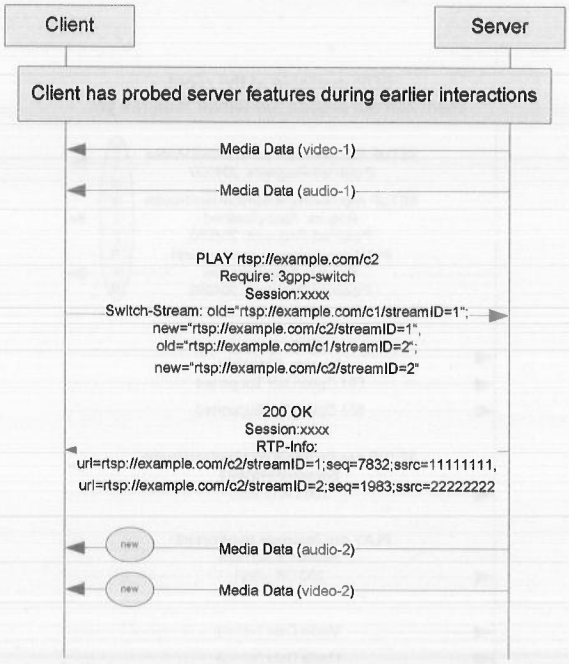
Note the first setup interaction is successful, since the client has not used the require header.



M.2 Content Switch with SDP

M.2.1 Successful Content Switch with available SDP

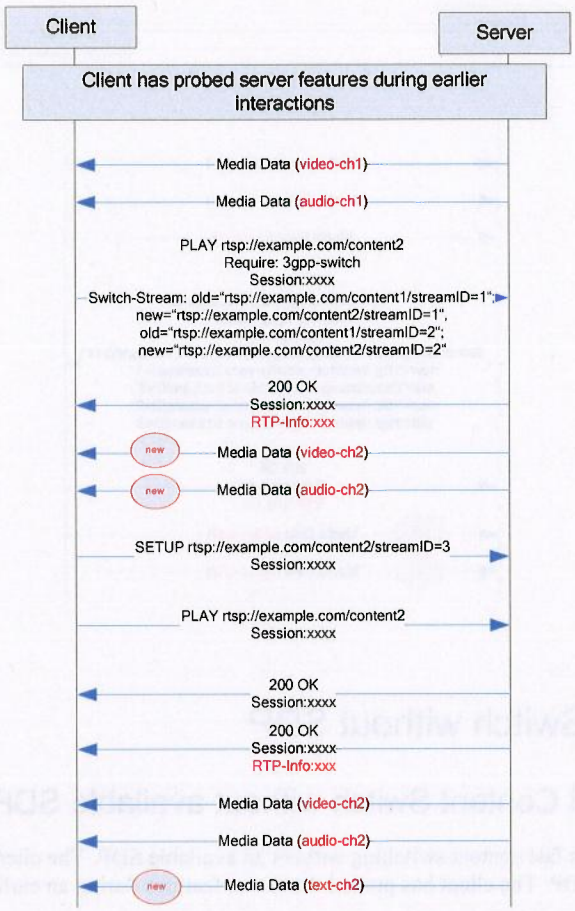
The following example depicts fast content switching with an available SDP. The client has retrieved the SDP prior to the content switch and has probed the server features during an earlier interaction.



M.2.2 Partial successful Content Switch with available SDP

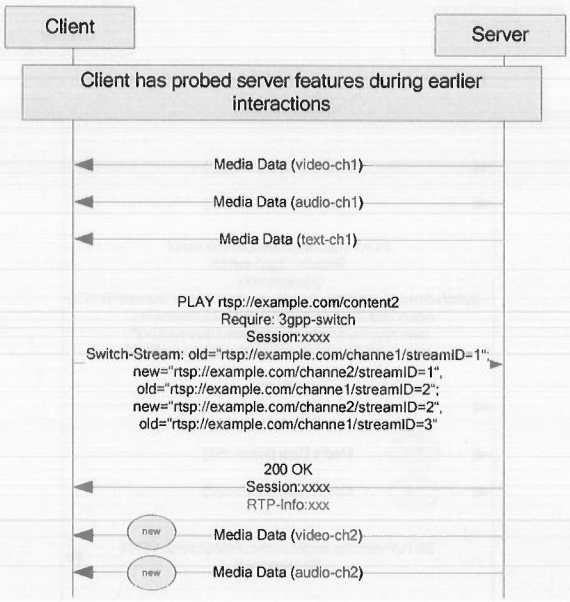
The following example depicts fast content switching with an available SDP. The content in the new SDP contains additional media components. The client has retrieved the SDP prior to the content switch and has probed the server features during an earlier interaction.

The client first switches the content to get the new data as quickly as possible. After that, the client adds a new media component.



M.2.3 Successful Content Switch with available SDP, but removal of media component

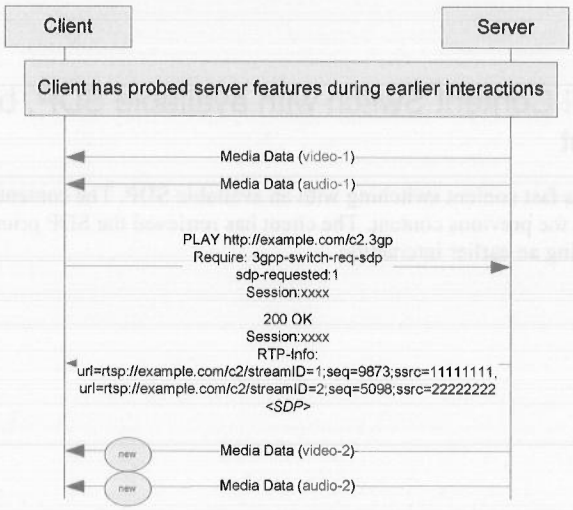
The following example depicts fast content switching with an available SDP. The content in the new SDP contains fewer media components than the previous content. The client has retrieved the SDP prior to the content switch and has probed the server features during an earlier interaction.



M.3 Content Switch without SDP

M.3.1 Successful Content Switch without available SDP

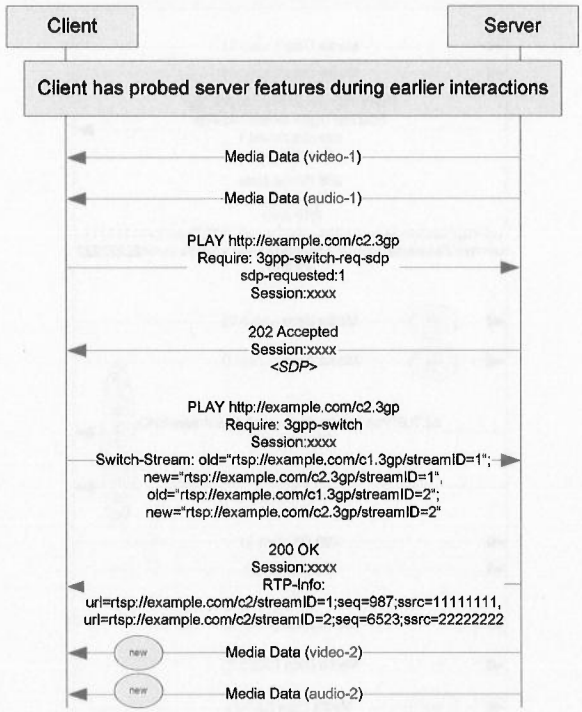
The following example depicts fast content switching without an available SDP. The client has received a content URL but has not yet retrieved the SDP. The client has probed the server features during an earlier interaction.



M.3.2 Partial successful Content Switch without available SDP

The following example depicts fast content switching without an available SDP. The client has only a content URL and still must retrieve the content description. The client has probed the server features during an earlier interaction.

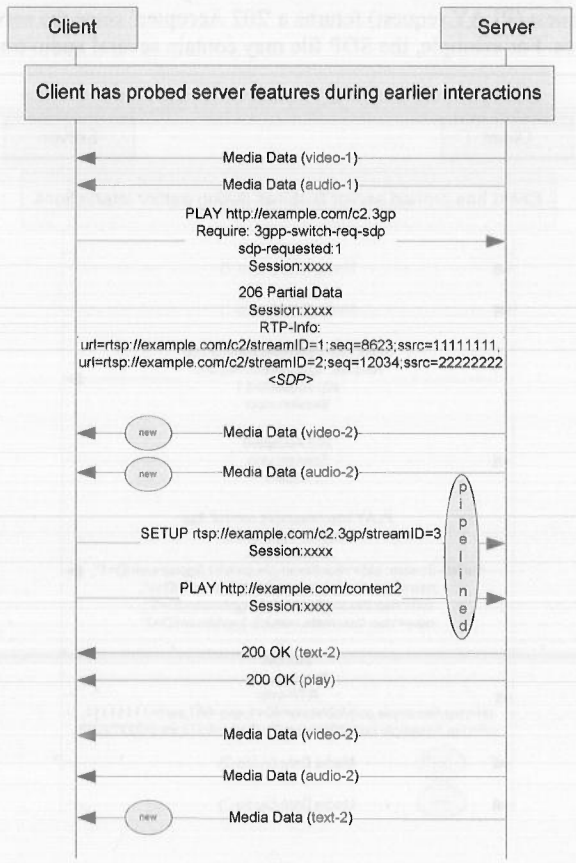
In this example, the switch request (PLAY request) returns a '202 Accepted' since the server was not able to select the desired flows from the SDP files. For example, the SDP file may contain several audio tracks and the server was unable to make a selection.



M.3.3 Partial successful Content Switch without available SDP

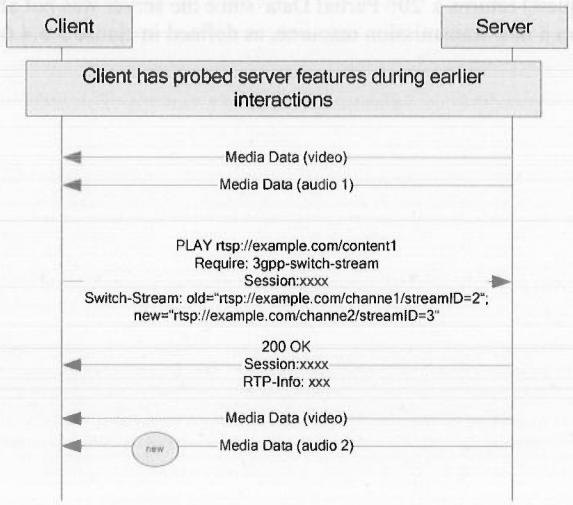
The following example depicts fast content switching without an available SDP. The client has only a content URL and still must retrieve the content description. The client has probed the server feature capabilities during an earlier interaction.

The switch request (PLAY request) returns a '206 Partial Data' since the server was not able to send all data flows of the SDP file. The client first sets up a new transmission resource, as defined in clause 5.5.4.6 (Addition of Media components)

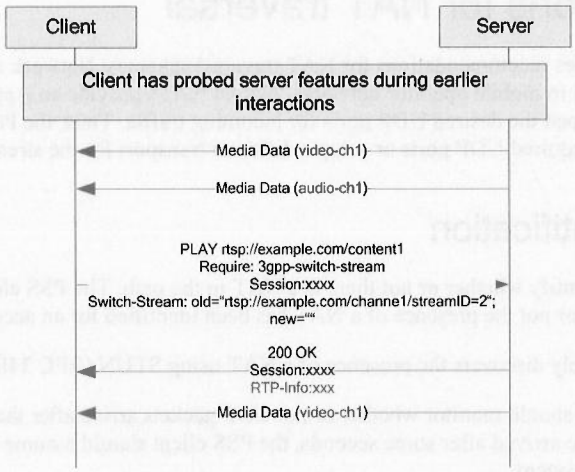


M.4 Stream Switching

M.4.1 Successful Stream Switch



M.4.2 Removal of Media Components from an ongoing session



Annex N (informative): Recommendations for NAT traversal

This informative annex provides recommendations for NAT traversal schemes. Network address translators are frequently used functions even in mobile operator networks. Not all NATs provide an Application Layer Gateway (ALG) for RTSP services to open the desired UDP ports for incoming traffic. Thus, the PSS client may need to use other techniques to open the required UDP ports or setup a different transport for the streaming media.

N.1 NAT identification

The PSS client should first identify whether or not there is a NAT in the path. The PSS client may, for future sessions, store the information whether or not the presence of a NAT has been identified for an access system.

Active Mode: The client actively discovers the presence of a NAT using STUN (RFC 3489).

Passive Mode: The PSS client should monitor whether or not UDP packets arrive after the RTSP PLAY request was issued. If no UDP packets have arrived after some seconds, the PSS client should assume the presence of a NAT/Firewall for this access system.

N.2 NAT traversal

If the PSS Client has identified the presence of a NAT, it may probe one or all of the following procedures.

UPnP (Universal Plug and Play) defines a set of procedures to discover gateways and open port-forwarding on client request. UPNP may not be implemented or activated in all NATs. Thus, the client should not expect UPnP present in all cases.

UDP Port Punching: With many existing NATs, the PSS client can initiate NAT routing by "punching" the UDP ports before packets are sent by the server. This is done after the port pairs are negotiated (via SETUP) and before streaming is initiated via PLAY, by sending an RTP packet on the RTP port pair and an RTCP report on the RTCP port pair.

RTSP/RTP Interleaving: The RTP data is interleaved with the RTSP data on a single TCP connection (defined in RFC 2326 section 10.12). RTSP/RTP Interleaving is implemented by a high variety of available streaming servers.

'RTP over TCP' (RFC4571): The RTP packets are tunnelled over separate TCP connections. A major difference compared to the 'RTSP/RTP Interleaving' mode (RFC 2326) is, that the client opens one or more separate TCP connections to the server for the RTP transport. This mechanism is described in the RTSP 2.0 draft (draft-ietf-mmusic-rfc2326bis-18.txt).

The PSS Client may also use other transports than RTP. For instance the PSS Client may also try to use progressive download as defined in clause 5.1.

Annex O (informative): Examples for PSS Timeshift

In the following, the PSS timeshift functionality is described using different examples.

In the following example, a PSS client starts the live session consumption in timeshift mode. This may happen when changing from MBMS reception to PSS reception. Note the PSS client uses the wallclock time from the MBMS stream to start the PSS session.

```
C->S:    PLAY rtsp://example.com/fizzle/foo RTSP/1.0
         CSeq: 835
         Session: 12345678
         Range: clock=20080401T072901.1Z-

S->C:    RTSP/1.0 200 OK
         CSeq: 835
         Range: clock=20080401T072901.1Z-
         3GPP-TS-CurrentRecording-Time: clock=20080401T072905.1Z
         3GPP-TS-Buffer: clock=20080401T062905.1Z-20080401T072905.1Z
```

The timeshift buffer in the example above is described by a closed ranged. The timeshift buffer is not progressing as sliding window.

In the following, the use of the 3GPP-TS-Buffer RTSP header is clarified. The 'client -> server' request messages left are not shown in these examples.

The server is maintaining a constant 3600 second buffer in the following example. The current time of the live session (at the time of response generation) is as specified in the CurrentRecording-Time.

```
S->C:    RTSP/1.0 200 OK
         CSeq: 835
         3GPP-TS-CurrentRecording-Time: clock=20080401T072905.1Z
         3GPP-TS-Buffer: buffer-depth=3600
```

The timeshift buffer is still being established and will progress as sliding window in the following example.

```
S->C:    RTSP/1.0 200 OK
         CSeq: 835
         3GPP-TS-CurrentRecording-Time: clock=20080401T072905.1Z
         3GPP-TS-Buffer: clock=20080401T062905.1Z-; buffer-depth=3600
```

Recording has started at 06:29:05.1, 1 April 2008 and will progress until a buffer-depth of 3600 seconds is reached; once 3600 seconds is reached then the timeshift buffering progresses in a sliding window. Once the PSS timeshift buffer is fully established, the PSS server should only give the 'buffer-depth' parameter with the 3GPP-TS-Buffer header.

The following example shows a static timeshift buffer. The buffer only contains information between the given closed range.

```
S->C:    RTSP/1.0 200 OK
         CSeq: 835
         3GPP-TS-CurrentRecording-Time: clock=20080401T072905.1Z
         3GPP-TS-Buffer: clock=20080401T062905.1Z-20080401T072905.1Z
```

Time shifting is available between the times specified (06:29:05.1 and 07:29:05.1, 1 April 2008). For example, this may be a recording of an earlier broadcast.

An open timeshift buffer range range is used in the following example

```
S->C:    RTSP/1.0 200 OK
         CSeq: 835
```

```
3GPP-TS-CurrentRecording-Time: clock=20080401T072905.1Z
3GPP-TS-Buffer: clock=20080401T062905.1Z-
```

Time shifting is available indefinitely, beginning from 06:29:05.1. For example, this may be a live event in progress which is being recorded in its entirety for time-shifting purposes.

Annex P (informative): QoE Reporting Management Object Device Description Framework

This Device Description Framework (DDF) is the standardized minimal set. A vendor can define its own DDF for the complete device. This DDF can include more features than this minimal standardized version.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE MgmtTree PUBLIC "-//OMA//DTD-DM-DDF 1.2//EN"
"http://www.openmobilealliance.org/tech/DTD/dm_ddf-v1_2.dtd">
<MgmtTree>
   <VerDTD>1.2</VerDTD>
   <Man>--The device manufacturer--</Man>
   <Mod>--The device model--</Mod>
   <Node>
      <NodeName>3GPP_PSSQOE</NodeName>
      <DFProperties>
         <AccessType>
            <Get/>
         </AccessType>
         <DFFormat>
            <node/>
         </DFFormat>
         <Occurrence>
            <ZeroOrOne/>
         </Occurrence>
         <Scope>
            <Permanent/>
         </Scope>
         <DFTitle>The interior node holding all 3GPP PSS QoE Metrics Reporting objects</DFTitle>
         <DFType>
            <DDFName/>
         </DFType>
      </DFProperties>
      <Node>
         <NodeName>Enabled</NodeName>
         <DFProperties>
            <AccessType>
               <Get/>
            </AccessType>
            <DFFormat>
               <bool/>
            </DFFormat>
            <Occurrence>
               <One/>
            </Occurrence>
            <Scope>
               <Permanent/>
            </Scope>
            <DFTitle>The QoE reporting requested indicator</DFTitle>
            <DFType>
               <DDFName/>
            </DFType>
         </DFProperties>
      </Node>
      <Node>
         <NodeName>Servers</NodeName>
         <DFProperties>
            <AccessType>
               <Get/>
            </AccessType>
            <DFFormat>
               <chr/>
            </DFFormat>
            <Occurrence>
               <One/>
            </Occurrence>
            <DFTitle>The URL of the QoE report servers</DFTitle>
            <DFType>
               <DDFName/>
            </DFType>
```

```
      </DFProperties>
   </Node>
   <Node>
      <NodeName>APN</NodeName>
      <DFProperties>
         <AccessType>
            <Get/>
         </AccessType>
         <DFFormat>
            <chr/>
         </DFFormat>
         <Occurrence>
            <ZeroOrOne/>
         </Occurrence>
         <DFTitle>The Access Point Name for QoE reporting</DFTitle>
         <DFType>
            <DDFName/>
         </DFType>
      </DFProperties>
   </Node>
   <Node>
      <NodeName>Format</NodeName>
      <DFProperties>
         <AccessType>
            <Get/>
         </AccessType>
         <DFFormat>
            <chr/>
         </DFFormat>
         <Occurrence>
            <ZeroOrOne/>
         </Occurrence>
         <DFTitle>The QoE metrics report format</DFTitle>
         <DFType>
            <DDFName/>
         </DFType>
      </DFProperties>
   </Node>
   <Node>
      <NodeName>Rules</NodeName>
      <DFProperties>
         <AccessType>
            <Get/>
         </AccessType>
         <DFFormat>
            <chr/>
         </DFFormat>
         <Occurrence>
            <ZeroOrOne/>
         </Occurrence>
         <DFTitle>The QoE metrics rules</DFTitle>
         <DFType>
            <DDFName/>
         </DFType>
      </DFProperties>
   </Node>
   <Node>
      <NodeName>Session</NodeName>
      <DFProperties>
         <AccessType>
            <Get/>
         </AccessType>
         <DFFormat>
            <node/>
         </DFFormat>
         <Occurrence>
            <ZeroOrOne/>
         </Occurrence>
         <DFTitle>The QoE session metrics node</DFTitle>
         <DFType>
            <DDFName/>
         </DFType>
      </DFProperties>
      <Node>
         <NodeName>Metrics</NodeName>
         <DFProperties>
            <AccessType>
               <Get/>
```

```
            </AccessType>
            <DFFormat>
               <chr/>
            </DFFormat>
            <Occurrence>
               <ZeroOrOne/>
            </Occurrence>
            <DFType>
               <DDFName/>
            </DFType>
         </DFProperties>
      </Node>
      <Node>
         <NodeName>Ext</NodeName>
         <DFProperties>
            <AccessType>
               <Get/>
            </AccessType>
            <DFFormat>
               <node/>
            </DFFormat>
            <Occurrence>
               <ZeroOrOne/>
            </Occurrence>
            <Scope>
               <Permanent/>
            </Scope>
            <DFTitle> A collection of all extension objects</DFTitle>
            <DFType>
               <DDFName/>
            </DFType>
         </DFProperties>
      </Node>
   </Node>
   <Node>
      <NodeName>Speech</NodeName>
      <DFProperties>
         <AccessType>
            <Get/>
         </AccessType>
         <DFFormat>
            <node/>
         </DFFormat>
         <Occurrence>
            <ZeroOrOne/>
         </Occurrence>
         <DFTitle>The QoE speech metrics node</DFTitle>
         <DFType>
            <DDFName/>
         </DFType>
      </DFProperties>
      <Node>
         <NodeName>Metrics</NodeName>
         <DFProperties>
            <AccessType>
               <Get/>
            </AccessType>
            <DFFormat>
               <chr/>
            </DFFormat>
            <Occurrence>
               <ZeroOrOne/>
            </Occurrence>
            <DFType>
               <DDFName/>
            </DFType>
         </DFProperties>
      </Node>
      <Node>
         <NodeName>Ext</NodeName>
         <DFProperties>
            <AccessType>
               <Get/>
            </AccessType>
            <DFFormat>
               <node/>
            </DFFormat>
            <Occurrence>
```

```
                <ZeroOrOne/>
            </Occurrence>
            <Scope>
                <Permanent/>
            </Scope>
            <DFTitle> A collection of all extension objects</DFTitle>
            <DFType>
                <DDFName/>
            </DFType>
        </DFProperties>
    </Node>
</Node>
<Node>
    <NodeName>Video</NodeName>
    <DFProperties>
        <AccessType>
            <Get/>
        </AccessType>
        <DFFormat>
            <node/>
        </DFFormat>
        <Occurrence>
            <ZeroOrOne/>
        </Occurrence>
        <DFTitle>The QoE video metrics node</DFTitle>
        <DFType>
            <DDFName/>
        </DFType>
    </DFProperties>
    <Node>
        <NodeName>Metric</NodeName>
        <DFProperties>
            <AccessType>
                <Get/>
            </AccessType>
            <DFFormat>
                <chr/>
            </DFFormat>
            <Occurrence>
                <ZeroOrMore/>
            </Occurrence>
            <DFType>
                <DDFName/>
            </DFType>
        </DFProperties>
    </Node>
    <Node>
        <NodeName>Ext</NodeName>
        <DFProperties>
            <AccessType>
                <Get/>
            </AccessType>
            <DFFormat>
                <node/>
            </DFFormat>
            <Occurrence>
                <ZeroOrOne/>
            </Occurrence>
            <Scope>
                <Permanent/>
            </Scope>
            <DFTitle>A collection of all extension objects</DFTitle>
            <DFType>
                <DDFName/>
            </DFType>
        </DFProperties>
    </Node>
</Node>
<Node>
    <NodeName>Text</NodeName>
    <DFProperties>
        <AccessType>
            <Get/>
        </AccessType>
        <DFFormat>
            <node/>
        </DFFormat>
        <Occurrence>
```

```
            <ZeroOrOne/>
         </Occurrence>
         <DFTitle>The QoE timed text metrics node</DFTitle>
         <DFType>
            <DDFName/>
         </DFType>
      </DFProperties>
      <Node>
         <NodeName>Metric</NodeName>
         <DFProperties>
            <AccessType>
               <Get/>
            </AccessType>
            <DFFormat>
               <chr/>
            </DFFormat>
            <Occurrence>
               <ZeroOrOne/>
            </Occurrence>
            <DFType>
               <DDFName/>
            </DFType>
         </DFProperties>
      </Node>
      <Node>
         <NodeName>Ext</NodeName>
         <DFProperties>
            <AccessType>
               <Get/>
            </AccessType>
            <DFFormat>
               <node/>
            </DFFormat>
            <Occurrence>
               <ZeroOrOne/>
            </Occurrence>
            <Scope>
               <Permanent/>
            </Scope>
            <DFTitle>A collection of all extension objects</DFTitle>
            <DFType>
               <DDFName/>
            </DFType>
         </DFProperties>
      </Node>
   </Node>
   <Node>
      <NodeName>Ext</NodeName>
      <DFProperties>
         <AccessType>
            <Get/>
         </AccessType>
         <DFFormat>
            <node/>
         </DFFormat>
         <Occurrence>
            <ZeroOrOne/>
         </Occurrence>
         <Scope>
            <Permanent/>
         </Scope>
         <DFTitle>A collection of all extension objects</DFTitle>
         <DFType>
            <DDFName/>
         </DFType>
      </DFProperties>
   </Node>
</Node>
</MgmtTree>
```

Annex Q (Informative): Guidelines for Adaptive HTTP Streaming

Q.1 Content-Preparation Modes

Q1.1 Introduction

The specification on adaptive HTTP Streaming is restricted to the interface between the HTTP -Streaming Client and the HTTP-Streaming Server. The content preparation on the network-side is out of scope of this specification. In this clause, guidelines on two different modes how the network can prepare the content to serve the HTTP requests issued by the HTTP-Streaming client.

Q.1.2 Static Mode

Static content preparation mode is an approach for delivering media content over HTTP as static content. The server is not required to prepare the content in any way. Instead, the content preparation is done in advance, possibly offline, by a separate entity. The server may be a web server that serves the media file(s) as any other regular static file.

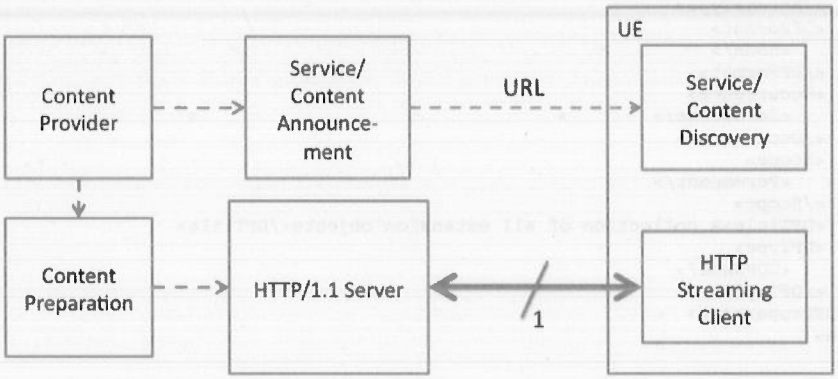


Figure Q.1 HTTP-Streaming Architecture for the Static Content Serving Mode

Q.1.3 Dynamic Mode

In dynamic content serving mode, the streaming server dynamically tailors the streamed content to a client based on requests from the client. The HTTP streaming server interprets the incoming HTTP GET request and identifies the requested media samples from a given content. The server then locates the requested media samples in the content file(s) or from the live stream. It then extracts and envelopes the requested media samples in a container. Subsequently, the newly formed container with the media samples is delivered to the client in the HTTP GET response body.

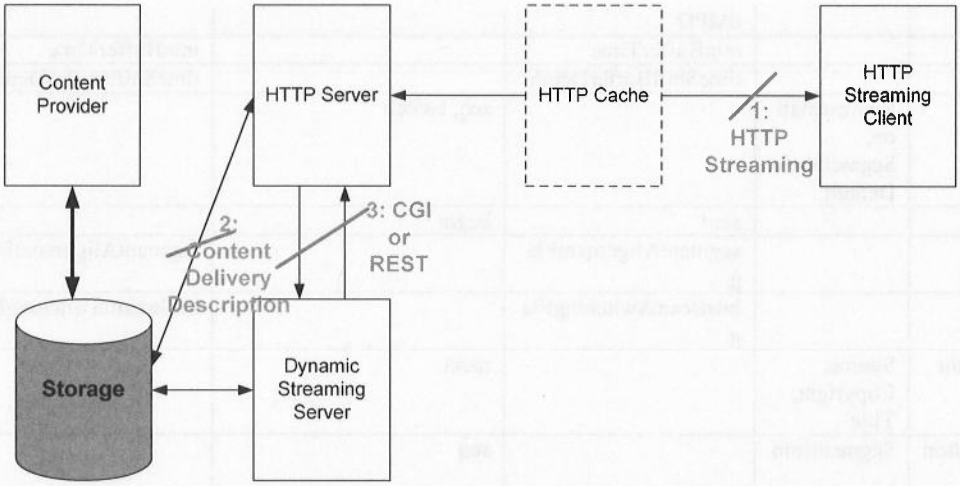


Figure Q.2 HTTP-Streaming Architecture for the Dynamic Content Serving Mode

Q.2 Mapping MPD structure and semantics to SMIL

Q.2.1 General

The mapping presented in this Annex allows transformation of the MPD table and the XML schema defined in section 12.2.5 to a SMIL-based syntax. This transformation may be effected automatically, for instance, using XSLT, at the client or the server. The MPD structure and semantics will be retained in the SMIL-based syntax.

The first 3 columns of Table 12.4 below contain the elements and attributes from the MPD table 0.2. Column 1 contains an MPD element, column 2 lists its children elements and column 3 lists its attributes.

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

Note that '3GPP SMIL' as used in this document refers to the 3GPP SMIL Language profile defined in TS 26.246 [52].

In some cases to match the semantics from Table 12.2, new attributes/elements that are not defined in 3GPP SMIL, are required. Column 5 lists these attributes or elements. These would be added to 3GPP SMIL as extensions indicated by the '3g9' identifier defined in the same namespace as that for 3GPP HTTP streaming in section 12.2.5, viz., xmlns="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009".

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

Table Q1: Mapping MPD structure, semantics and syntax to SMIL

Element	Children Elements	Attribute	Mapping to 3GPP SMIL	Extension to 3GPP SMIL
MPD	Period, ProgramInformation		body	
		type		type
		availabilityStartTime		availabilityStartTime
		availabilityEndTime		availabilityEndTime
		duration		duration
		minimumUpdatePerio		minimumUpdatePeriodMPD

		dMPD		
		minBufferTime		minBufferTime
		timeShiftBufferDepth		timeShiftBufferDepth
Period	Representation, SegmentInfoDefault		seq, switch	
		start	begin	
		segmentAlignmentFlag		segmentAlignmentFlag
		bitstreamSwitchingFlag		bitStreamSwitchingFlag
ProgramInformation	Source, Copyright, Title		meta	
Representation	SegmentInfo, ContentProtection, TrickMode		seq	
		id	id	
		bandwidth	systemBitrate	
		width, height	systemScreenSize	
		language	systemLanguage	
		mimeType	systemComponent	
		startWithRAP		startWithRAP
		qualityRanking		qualityRanking
SegmentInfo	Initialisation SegmentUrl, Url, UrlTemplate		par, seq When track alignment across Segments cannot be guaranteed, par should be used with each children URL containing begin and dur attributes.	SegmentInfo element with playback semantics identical to those defined in section 12.2. That is, all children elements of SegmentInfo are time-continuous across boundaries of consecutive Media Segments within one Representation.
		duration	dur specified for each Url in playlist, possibly in conjunction with begin	
		baseURL		baseURL
InitSegmentUrl			See Url below. To identify initialization segments, either dur can be set to '0' or type can be set to 'init'	
Url			MEDIA-ELMS (ref, video, audio, etc.) as defined in 3GPP SMIL along with all attributes defined for those elements	
		sourceURL	src (only allows absolute URIs).	sourceURL with URI resolution semantics defined in 12.2.4.2.1. This attribute is also defined for the MPD, SegmentInfo, SegmentInfoDefault and Url elements
		range		range
UrlTemplate			Url playlists may be used as an alternative to urlTemplate	UrlTemplate (along with attributes defined for UrlTemplate)

SegmentInfoDefault	UrlTemplate		This element may be skipped and information provided directly at SegmentInfo level instead	SegmentInfoDefault
		duration		See duration in Segment
ContentProtection	SchemeInfo		For 3GP files, content protection may be achieved through mechanisms defined in 3GP file format	ContentProtection
		schemeId		schemeId
SchemeInformation				SchemeInfo
TrickMode	alternatePlayoutMode			TrickMode

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

Q.2.2 Examples

Q.2.2.1 Example 1: MPD for on-demand content with multiple Periods and alternate Representations

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

<!-- Period 1-->
<seq begin='0s'>
<!-- alternate set of Representations available during this Period -->
    <switch>
        <!-- low bitrate  Representation with 15-sec Segments -->
        <seq systemBitrate='128000'>
            <video dur='0s'  src="http://server/path/rep1/clip_init.3gp"/>
            <par>
                    <video begin='0s' dur='15s' src="http://server/path/rep1/clip_1.3gp"/>
                    <video begin='15s' dur='15s' src="http://server/path/rep1/clip_1.3gp"/>
                    ...
                    <video begin='585s' dur='15s' src="http://server/path/rep1/clip_40.3gp"/>
            </par>
        </seq>
        <!-- mid bitrate Representation with 30-sec Segments -->
        <seq systemBitrate='256000'>
            <video dur='0s' src="http://server/path/rep2/clip_init.3gp'/>
            <par>
                    <video begin='0s' dur='30s' src="http://server/path/rep2/clip_1.3gp"/>
                    ...
                    <video begin='570s' dur='30s' src="http://server/path/rep2/clip_20.3gp"/>
            </par>
        </seq>
        <!-- high bitrate Representation with 30-sec Segments-->
        <seq systemBitrate='512000'>
            <video type='init' dur='0s' src="http://server/path/rep3/clip_init.3gp'/>
            <par>
                    <video begin='0s' dur='30s' src="http://server/path/rep3/clip_1.3gp"/>
                    ...
                    <video begin='570s' dur='30s' src="http://server/path/rep3/clip_20.3gp"/>
            </par>
        </seq>
    </switch>
</seq> <!-- end of Period 1 -->

<!-- Period 2 begins 10 minutes after presentation start -->
<seq begin='600s'>
    <switch>
        <!-- english ad -->
        <seq systemLanguage='en' >
```

```
            <seq>
                    <video src='http://adserver/getad.php?id=1'/>
            </seq>
        </seq>
        <!-- french ad -->
        <seq systemLanguage='fr' >
            <seq>
                    <video src='http://adserver/getad.php?id=2'/>
            </seq>
        </seq>
    </switch>
</seq>

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

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

Q.2.2.2 Example 2: MPD for live content.

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

```
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
      xmlns:3g9="urn:3GPP:ns:PSS:AdaptiveHTTPStreamingMPD:2009">
<body 3g9: minimumUpdatePeriodMPD ='120s' 3g9:availabilityStartTime='2010-01-27T13:00Z'
3g9:availabilityEndTime='2010-01-27T15:00Z'3g9:minBufferTime='10s' 3g9:type='live'>
<!-- start of a Period -->
<seq begin='0s'>
    <!-- a single Representation -->
    <seq>
        <par>
                <video begin='0s' dur='10s' src='http://server/live_clip_1.m2ts'/>
                …
                <video begin='110s' dur='10s' src='http://server/live_clip_12.m2ts'/>
        </par>
    </seq> <!-- end of Representation -->
  </seq> <!-- end of Period -->
</body>
</smil>
```

Annex R (normative): Content Protection extensions

This annex specifies extensions to support protected PSS streaming, typically in conjunction with a Digital Rights Management system. The following is defined in the following clauses:

- the use of ISMACryp for transport of confidentiality protected PSS streams
- the optional use of OMA BCAST Short-Term Key Messages for carriage of frequently changing keys over UDP
- SDP signalling for the use of protected PSS streaming

R.1 Encryption and Transport of Protected PSS Streams

R.1.1 Overview

Streaming of content protected PSS streams according to this specification uses ISMACryp 2.0 [102] which is backward compatible to ISMACryp 1.1 [101]. Streaming content is either directly encrypted using content keys provided by a DRM system, or optionally with traffic keys transported in an additional key stream. This specification defines a mapping for traffic keys transported using the OMA BCAST Short Term Key Messages (STKM) format [103].

R.1.2 Stream format

RTP streaming of PSS content protected according to this specification shall use ISMACryp 2.0 [102] as detailed further in this clause, i.e. by encrypting elementary audio and video Access Units (AUs).Each encrypted AU has an ISMACrypContextAU as defined in [102].

R.1.3 Encryption

R.1.3.1 Encryption Algorithm

The encryption algorithm shall be AES (Advanced Encryption Standard) with 128 bit key size in counter mode, i.e., AES_CTR_128 as defined in [102]. Note that this is the default cipher mode of ISMACryp and that it is not recommended to signal it in the SDP according to [102]. Other encryption algorithms, key sizes or chaining modes shall not be used.

Note that ISMACryp counter mode increases the counter for each byte of data by one, i.e., the counter is increased by 16 for each block of 128 bits.

To allow for storage in file formats that do not support a salt key, the ISMACryp salt key k_s [102] should be zero. Note that this is the default value and that it is not recommended to signal this in the SDP.

R.1.3.2 Content encryption using a single key

All individual AUs of a PSS stream may be encrypted directly with a single content key, provided by a DRM system [out of scope of this specification] or alternatively signalled within the SDP, and used as encryption key according to [102]. A client implementing this specification shall support direct encryption of the AUs with a single content key. If no key stream is signalled in the SDP for a media stream, the client shall assume that direct encryption with a single content key is used.

For direct encryption, ISMACrypKeyIndicatorLength shall be equal to zero. Note that this is the default value for ISMACryp and that it is not recommended to signal it in the SDP according to [102].

R.1.3.3 Content encryption using a key stream

AUs may be encrypted with Traffic Keys transported in Key Messages in an additional key stream. Support for this mode is optional for clients implementing this specification. Key Messages, if used, shall use the OMA BCAST Short Term Key Messages (STKM) format for the OMA BCAST DRM profile as described in Section 5.5 of [103], with the following restrictions:

- traffic_protection_protocol shall be set to TKM_ALGO_ISMACRYP
- traffic_authentication_flag should be set to TKM_FLAG_FALSE
- access_criteria_flag should be set to TKM_FLAG_FALSE
- program_flag should be set to TKM_FLAG_FALSE
- service_flag should be set to TKM_FLAG_FALSE

Traffic Keys may be transmitted in the encrypted_traffic_key_material and next_encrypted_traffic_key_material fields as defined in Section 5.5 of [103]. It shall be indicated for each AU in the ISMACrypContextAU header which Traffic Key was used to encrypt this AU as specified in [103]. If OMA BCAST Short Term Key Messages are used, the keys in the STKM shall be encrypted with the content key from the DRM system.

OMA BCAST Key Messages, if used, are delivered in a separate UDP stream as specified in Section 5.5 of [103]. Key Messages are associated to ISMACryp streams via SDP signalling.

R.1.4 RTP Transport of Encrypted AUs

Content encryption modifies data before packetization of RTP packets, thus the various RFCs defining ways to encapsulate audio and video data do not apply. In addition, some signalling is necessary in the SDP in order to enable the decryption of the data. ISMACryp 1.1 [101] has defined encapsulation for some MPEG-4 codecs. For these codecs, the encapsulation as defined in [101] shall be used. For any other encrypted media that has a defined mapping to the ISO Media File Format, the encapsulation as defined in section 7 of [102] shall be used.

R.1.5 RTSP Signalling for key stream (STKM) setup and control

If STKMs are used as described in clause R.1.3.3, the RTSP session setup also needs to establish the STKM session, and a control URI as defined in [5] shall be present for each STKM media description in the SDP description.

R.1.5.1 RTSP SETUP Method

The control URI is used within the RTSP SETUP method to establish the described STKM sessions.

The RTSP transport protocol specifier for STKM as defined in [5] shall be "vnd.oma.bcast.stkm/UDP". One and only one UDP port is allocated for each STKM channel.

The following RTP specific parameters shall be used in the transport request and responds header for STKM sessions:

- client_port: This parameter provides the unicast STKM port(s) on which the client has chosen to receive STKM data.
- server_port: This parameter provides the unicast STKM port(s) on which the server has chosen to send data.

R.1.5.2 RTSP PLAY, PAUSE and TEARDOWN Method

The PLAY method tells the server to start sending data including STKM session data as defined in [5].

The PAUSE request causes the stream delivery including all STKM sessions to be interrupted (halted) as defined in [5].

The TEARDOWN client to server request stops the stream delivery including all STKM data delivery for the given URI, freeing the resources associated with it. Details for the TEARDOWN method are defined in [5].

R.2 SDP Signalling

SDP signalling of protected PSS streams shall be done as described in section 8 of [102].

NOTE: the mentioned section describes how to signal crypto suite (please note the corresponding remark in clause R.1.3.1), IV length, key indicator length suite (please note the corresponding remark in clause R.1.3.2), selective encryption, salt key suite (please note the corresponding remark in clause R.1.3.1), key management system, key, delta IV length, and key indicator per AU, using `fmtp` attributes. The signalling of key management system allows associating the protected PSS streams with a DRM system, and thus enables the PSS player to contact the DRM agent, in order to handle stream decryption.

Additionally, DRM system specific parameters may be signalled in the SDP, but are out of scope for the normative part of this Annex (they are some example protection systems discussed in clause R.4). If a PSS client does not understand such parameters, it should ignore them.

If key streams using STKM stream format are used, SDP signalling of key streams shall be done as specified in sections 10.1.2 and 10.1.3 of [103], with the following restrictions:

- streamid parameter shall be included for SDP signalling of key streams; all other parameters mentioned in 10.1.2 of [103] are optional
- stkmstream parameter (section 10.1.3 of [103]) shall be included for SDP signalling of media streams and be used to link media streams and STKM streams

R.3 Enforcement of permissions and constraints

If a DRM system is used, it may specify usage rights (permissions and constraints) in the license that also contains the content key. How these rights are enforced is an implementation issue and out of scope of this specification.

R.4 Mapping to DRM systems (informative)

R.4.1 Mapping to OMA DRM 2.0 (informative)

If this Annex is used in conjunction with OMA DRM 2.0 based key management, the ISMACrypKMSID parameter in the SDP is set to 'odkm' and the ISMACrypKMSVersion is set to '0x0000200". ISMACrypKMSSpecificData may be used to signal OMA DRM 2.0 specific parameters, e.g. RI URL or Silent URL. It should at least be used to signal the Content ID.

R.4.2 Mapping to OMA DRM 2.1 (informative)

If this Annex is used in conjunction with OMA DRM 2.1 based key management, the ISMACrypKMSID parameter in the SDP is set to 'odkm' and the ISMACrypKMSVersion is set to '0x0000201". ISMACrypKMSSpecificData may be used to signal OMA DRM 2.1 specific parameters, e.g. RI URL or Silent URL. It should at least be used to signal the Content ID.

Annex S (normative): MIME Type Registrations

S.1 MIME Type Registration for Media Presentation Description

Type name: video

Subtype name: vnd.3gpp.mpd

Required parameters:

None.

Optional parameters:

None.

Encoding considerations:

None.

Security considerations:

A Denial-of-Service attack could be done by supplying MPDs with a very low minimum update period. As an MPD references external media, it is possible that they point to harmful media files. As MPDs may be unsigned, unencrypted and unhashed, they may be susceptible to a man-in-the-middle exploits.

Interoperability considerations:

None.

Published specification:

3GPP TS 26.234, Release 9.

Applications which use this media type:

Third Generation Partnership Project (3GPP) Adaptive HTTP Streaming.

Additional information:

Magic number(s):

None

File extension(s):

3gm

Person & email address to contact for further information:

John Meredith (john.meredith@etsi.org)

Intended usage:

Common

Restrictions on usage:

Author:

3GPP TSG SA WG4

Change controller:

3GPP TSG SA

Annex T (informative): Change history

Change history							
Date	TSG SA #	TSG Doc.	CR	Rev	Subject/Comment	Old	New
03-2001	11	SP-010094			Version for Release 4		4.0.0
09-2001	13	SP-010457	001	1	3GPP PSS4 SMIL Language Profile	4.0.0	4.1.0
09-2001	13	SP-010457	002		Clarification of H.263 baseline settings	4.0.0	4.1.0
09-2001	13	SP-010457	003	2	Updates to references	4.0.0	4.1.0
09-2001	13	SP-010457	004	1	Corrections to Annex A	4.0.0	4.1.0
09-2001	13	SP-010457	005	1	Clarifications to chapter 7	4.0.0	4.1.0
09-2001	13	SP-010457	006	1	Clarification of the use of XHTML Basic	4.0.0	4.1.0
12-2001	14	SP-010703	007		Correction of SDP Usage	4.1.0	4.2.0
12-2001	14	SP-010703	008	1	Implementation guidelines for RTSP and RTP	4.1.0	4.2.0
12-2001	14	SP-010703	009		Correction to media type decoder support in the PSS client	4.1.0	4.2.0
12-2001	14	SP-010703	010		Amendments to file format support for 26.234 release 4	4.1.0	4.2.0
03-2002	15	SP-020087	011		Specification of missing limit for number of AMR Frames per Sample	4.2.0	4.3.0
03-2002	15	SP-020087	013	2	Removing of the reference to TS 26.235	4.2.0	4.3.0
03-2002	15	SP-020087	014		Correction to the reference for the XHTML MIME media type	4.2.0	4.3.0
03-2002	15	SP-020087	015	1	Correction to MPEG-4 references	4.2.0	4.3.0
03-2002	15	SP-020087	018	1	Correction to the width field of H263SampleEntry Atom in Section D.6	4.2.0	4.3.0
03-2002	15	SP-020087	019		Correction to the definition of "b=AS"	4.2.0	4.3.0
03-2002	15	SP-020087	020		Clarification of the index number's range in the referred MP4 file format	4.2.0	4.3.0
03-2002	15	SP-020087	021		Correction of SDP attribute 'C='	4.2.0	4.3.0
03-2002	15	SP-020173	023		References to "3GPP AMR-WB codec" replaced by "ITU-T Rec. G.722.2" and "RFC 3267"	4.2.0	4.3.0
03-2002	15	SP-020088	022	2	Addition of Release 5 functionality	4.3.0	5.0.0
06-2002	16	SP-020226	024	1	Correction to Timed Text	5.0.0	5.1.0
06-2002	16	SP-020226	026	3	Mime media type update	5.0.0	5.1.0
06-2002	16	SP-020226	027		Corrections to the description of Sample Description atom and Timed Text Format	5.0.0	5.1.0
06-2002	16	SP-020226	029	1	Corrections Based on Interoperability Issues	5.0.0	5.1.0
09-2002	17	SP-020439	030	2	Correction regarding support for Timed Text	5.1.0	5.2.0
09-2002	17	SP-020439	032	3	Required RTSP header support	5.1.0	5.2.0
09-2002	17	SP-020439	034	1	Including bitrate information for H.263	5.1.0	5.2.0
09-2002	17	SP-020439	035	1	RTCP Reports and Link Aliveness in Ready State	5.1.0	5.2.0
09-2002	17	SP-020439	036	2	Correction on media and session-level bandwidth fields in SDP	5.1.0	5.2.0
09-2002	17	SP-020439	037	2	Correction on usage of MIME parameters for AMR	5.1.0	5.2.0
09-2002	17	SP-020439	038	1	Correction of Mapping of SDP parameters to UMTS QoS parameters (Annex J)	5.1.0	5.2.0
12-2002	18	SP-020694	039	2	Addition regarding IPv6 support in SDP	5.2.0	5.3.0
12-2002	18	SP-020694	040		Code points for H.263	5.2.0	5.3.0
12-2002	18	SP-020694	041	2	File format 3GP based on ISO and not MP4	5.2.0	5.3.0
12-2002	18	SP-020694	044	1	SMIL authoring instructions	5.2.0	5.3.0
12-2002	18	SP-020694	045	1	Client usage of bandwidth parameter at the media level in SDP	5.2.0	5.3.0
12-2002	18	SP-020694	047	1	SMIL Language Profile	5.2.0	5.3.0
12-2002	18	SP-020694	050	1	Usage of Multiple Media Sample Entries in Media Tracks of 3GP files	5.2.0	5.3.0
12-2002	18	SP-020694	051	1	Progressive download of 3GP files	5.2.0	5.3.0
03-2003	19	SP-030091	052	1	SDP bandwidth modifier for RTCP bandwidth	5.3.0	5.4.0
03-2003	19	SP-030091	053		Specification of stream control URLs in SDP files	5.3.0	5.4.0

03-2003	19	SP-030091	054		Clarification of multiple modifiers for timed text	5.3.0	5.4.0
03-2003	19	SP-030091	056	4	Correction of wrong references	5.3.0	5.4.0
03-2003	19	SP-030091	057	2	Correction of signalling frame size for H.263 in SDP	5.3.0	5.4.0
06-2003	20	SP-030217	058	1	SMIL supported event types	5.4.0	5.5.0
06-2003	20	SP-030217	060		Correction to the Content Model of the SMIL Language Profile	5.4.0	5.5.0
09-2003	21	SP-030448	061	1	Correction on session bandwidth for RS and RR RTCP modifiers	5.5.0	5.6.0
09-2003	21	SP-030448	062	1	Correction of ambiguous range headers in SDP	5.5.0	5.6.0
09-2003	21	SP-030448	063	1	Timed-Text layout example	5.5.0	5.6.0
09-2003	21	SP-030448	064		Correction of ambiguity in RTP timestamps handling after PAUSE/PLAY RTSP requests	5.5.0	5.6.0
09-2003	21	SP-030448	065		Correction of obsolete RTP references	5.5.0	5.6.0
09-2003	21	SP-030448	066	1	Correction of wrong reference	5.5.0	5.6.0
09-2003	21	SP-030448	067		Missing signaling of live content	5.5.0	5.6.0
06-2004	24	SP-040434	068	1	Addition of Release-6 functionality	5.6.0	6.0.0
09-2004	25	SP-040652	070	1	Additional Release-6 updates to PSS Protocols and codecs	6.0.0	6.1.0
09-2004	25	SP-040642	074	1	Introduction of Extended AMR-WB and Enhanced aacPlus into PSS service	6.0.0	6.1.0
09-2004	25	SP-040656	075	1	Introduction of the H.264 (AVC) video codec into the PSS service	6.0.0	6.1.0
12-2004	26	SP-040839	076		Correction of RDF schema for PSS capability vocabulary	6.1.0	6.2.0
12-2004	26	SP-040839	077		Transport-independent SDP bandwidth modifiers for PSS	6.1.0	6.2.0
12-2004	26	SP-040839	078		Correction of MIME type definition for DRM protected content	6.1.0	6.2.0
12-2004	26	SP-040839	079	1	Adoption of SVG Tiny 1.2 for PSS	6.1.0	6.2.0
03-2005	27	SP-050093	081		Correction to 26.234 NADU 'NUN' field regarding MPEG4 Video	6.2.0	6.3.0
03-2005	27	SP-050093	083		Correction of RDF schema for UAProf	6.2.0	6.3.0
03-2005	27	SP-050093	084		Correction of syntax and references	6.2.0	6.3.0
05-2005	28	SP-050248	085		Correction to QoE metrics specification for PSS	6.3.0	6.4.0
09-2005	29	SP-050427	0086		Correction to QoE metrics specification for PSS	6.4.0	6.5.0
09-2005	29	SP-050427	0087		Addition of QoE Metrics to UAPROF specification for PSS	6.4.0	6.5.0
09-2005	29	SP-050427	0088	1	Correction of SDP bandwidth modifiers	6.4.0	6.5.0
12-2005	30	SP-050787	0089	2	Correction of the SVGT1.2 content creation guideline	6.5.0	6.6.0
03-2006	31	SP-060010	0090	1	Clarification on the Use of Target-Time in 3GPP Adaptation Header	6.6.0	6.7.0
03-2006	31	SP-060010	0092		Correction of references and PSS capability vocabulary	6.6.0	6.7.0
03-2006	31	SP-060010	0093	1	Addition of a reference to TR 26.936	6.6.0	6.7.0
03-2006	31	SP-060010	0094		Correction of SDP attribute definition for asset information	6.6.0	6.7.0
06-2006	32	SP-060355	0095	1	Correction of references and signalling of UAProf profiles in PSS	6.7.0	6.8.0
06-2006	32	SP-060355	0096		Correction of reference to UAProf	6.7.0	6.8.0
09-2006	33	SP-060594	0097	4	Correction to unit discrepancy in GBW and MBW parameters of 3GPP-Link-Char header and other essential clarifications	6.8.0	6.9.0
09-2006	33	SP-060594	0098		Correction of references in PSS	6.8.0	6.9.0
09-2006	33	SP-060600	0099	3	Restrict size of audioMuxElements	6.9.0	7.0.0
12-2006	34	SP-060847	0101	1	Correction of Media Type to enable IANA registration	7.0.0	7.1.0
12-2006	34	SP-060847	0103		Correction of References	7.0.0	7.1.0
03-2007	35	SP-070030	0105	1	Clarification of SDP rtpmap rate and audio sampling frequency signalling	7.1.0	7.2.0
03-2007	35	SP-070028	0106	2	Upgrade of video codec requirements for PSS	7.1.0	7.2.0
06-2007	36	SP-070320	0107	5	Fast Content Switching and Start-up	7.2.0	7.3.0
06-2007	36	SP-070314	0110	1	Correction of references in PSS	7.2.0	7.3.0
06-2007	36	SP-070319	0111	1	Inclusion of DIMS in PSS	7.2.0	7.3.0

06-2007	36	SP-070320	0112	1	Content switch time QoE metric	7.2.0	7.3.0
06-2007	36	SP-070320	0113		PSSe QoE negotiation modifications	7.2.0	7.3.0
					Correction of ToC	7.3.0	7.3.1
09-2007	37	SP-070627	0115	1	Rewrite of language relating to Enhanced aacPlus signalling and RFC 3016	7.3.1	7.4.0
09-2007	37	SP-070627	0117	1	Correction to ABNF syntax of QoE metrics in PSS	7.3.1	7.4.0
03-2008	39	SP-080007	0118	2	Correction of PSS Fast Content Switching and start-up sections	7.4.0	7.5.0
03-2008	39	SP-080007	0120	1	Correction to SDP bandwidth specifiers in PSS	7.4.0	7.5.0
03-2008	39	SP-080007	0121	1	Clarifications to 3gpp-adaptation	7.4.0	7.5.0
09-2008	41	SP-080470	0123	1	'Fast Content Switching" corrections	7.5.0	7.6.0
09-2008	41	SP-080477	0125	1	Using Entity-Tags to ensure SDP validity during Fast Content Switching	7.6.0	8.0.0
09-2008	41	SP-080477	0126		Informative Annex on PSS NAT traversal	7.6.0	8.0.0
12-2008	42	SP-080681	0127	3	Introduction of the support for scaled playout	8.0.0	8.1.0
12-2008	42	SP-080681	0128	1	Introduction of PSS Timeshifting Functionality	8.0.0	8.1.0
03-2009	43	SP-090014	0129	2	PSS Timeshift Corrections	8.1.0	8.2.0
03-2009	43	SP-090006	0131	1	Corrections of Fast Content Switching and fast startup	8.1.0	8.2.0
03-2009	43	SP-090006	0134		Error in SDP for QoE configuration	8.1.0	8.2.0
06-2009	44	SP-090248	0136	2	Corrections to Fast Content Switching and QoE	8.2.0	8.3.0
09-2009	45	SP-090567	0137	2	Clean-up corrections	8.3.0	8.4.0
09-2009	45	SP-090562	0140	2	Correction of an FCS pipelining incompatibility when using Firewall packets	8.3.0	8.4.0
09-2009	45	SP-090572	0138	3	QoE Alignment for PSS	8.4.0	9.0.0
12-2009	46	SP-090710	0145	2	Adaptive HTTP Streaming in PSS	9.0.0	9.1.0
12-2009	46	SP-090710	0146	1	PSS DDF for QoE	9.0.0	9.1.0
12-2009	46	SP-090710	0147	1	PSS QoE reporting during buffering periods	9.0.0	9.1.0
12-2009	46	SP-090711	0148	2	Video profile and level updates	9.0.0	9.1.0
12-2009	46	SP-090710	0150	1	Update of Digital Rights Management Extensions for PSS	9.0.0	9.1.0
03-2010	47	SP-100024	0151	1	QoE Reporting for HTTP Streaming	9.1.0	9.2.0
03-2010	47	SP-100024	0157	1	Corrections to Content Protection extensions	9.1.0	9.2.0
03-2010	47	SP-100024	0158		Removal of obsolete DRM extensions annex K	9.1.0	9.2.0
03-2010	47	SP-100025	0165	2	Clarification about Main Profile	9.1.0	9.2.0
03-2010	47	SP-100024	0166	4	Updates to Adaptive HTTP Streaming	9.1.0	9.2.0
06-2010	48	SP-100375	0169	3	Essential Corrections to 3GPP Adaptive HTTP Streaming	9.2.0	9.3.0
06-2010	48	SP-100303	0170		Addition of Timed Graphics to PSS	9.2.0	9.3.0
06-2010	48	SP-100301	0171		Correction of ISMACrypKMSID parameter	9.2.0	9.3.0

History

Document history		
V9.1.0	January 2010	Publication
V9.2.0	April 2010	Publication
V9.3.0	June 2010	Publication