ETSI TS 126 234 V9.3.0 (2010-06)

Technical Specification

**Universal Mobile Telecommunications System (UMTS);
LTE;
Transparent end-to-end Packet-switched
Streaming Service (PSS);
Protocols and codecs
(3GPP TS 26.234 version 9.3.0 Release 9)**

Reference
RTS/TSGS-0426234v930

Keywords
LTE, UMTS

ETSI

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

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

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

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Intellectual Property Rights

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

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

Foreword

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

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

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

Contents

Foreword

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

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

Version x.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 or greater indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the specification;

The 3GPP transparent end-to-end packet-switched streaming service (PSS) specification consists of six 3GPP TSs: 3GPP TS 22.233 [1], 3GPP TS 26.233 [2], 3GPP TS 26.244 [50], 3GPP TS 26.245 [51], 3GPP TS 26.246 [52] and the present document.

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

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

Introduction

Streaming refers to the ability of an application to play synchronised media streams like audio and video streams in a continuous way while those streams are being transmitted to the client over a data network.

Applications, which can be built on top of streaming services, can be classified into on-demand and live information delivery applications. Examples of the first category are music and news-on-demand applications. Live delivery of radio and television programs are examples of the second category.

The 3GPP PSS provides a framework for Internet Protocol (IP) based streaming applications in 3G networks.

1 Scope

The present document specifies the protocols and codecs for the PSS within the 3GPP system. Protocols for control signalling, capability exchange, media transport, rate adaptation and protection are specified. Codecs for speech, natural and synthetic audio, video, still images, bitmap graphics, vector graphics, timed text and text are specified.

The present document is applicable to IP-based packet-switched networks.

2 References

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

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

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

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

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

[4] (void)

[5] IETF RFC 2326: "Real Time Streaming Protocol (RTSP)", Schulzrinne H., Rao A. and Lanphier R., April 1998.

[6] IETF RFC 4566: "SDP: Session Description Protocol", Handley M., Jacobson V. and Perkins C., July 2006.

[7] IETF STD 0006: "User Datagram Protocol", Postel J., August 1980.

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

[9] IETF RFC 3550: "RTP: A Transport Protocol for Real-Time Applications", Schulzrinne H. et al., July 2003.

[10] IETF RFC 3551: "RTP Profile for Audio and Video Conferences with Minimal Control", Schulzrinne H. and Casner S., July 2003.

[11] IETF RFC 4867: "RTP Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) Adaptive Multi-Rate Wideband (AMR-WB) Audio Codecs", Sjoberg J. et al., April 2007.

[12] (void)

[13] IETF RFC 3016: "RTP Payload Format for MPEG-4 Audio/Visual Streams", Kikuchi Y. et al., November 2000.

[14] IETF RFC 4629: "RTP Payload Format for the ITU-T Rec. H.263 Video", Ott J. et al., January 2007.

[15] IETF RFC 2046: "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", Freed N. and Borenstein N., November 1996.

[16] IETF RFC 3236: "The 'application/xhtml+xml' Media Type", Baker M. and Stark P., January 2002.

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

[18] 3GPP TS 26.071: "Mandatory Speech CODEC speech processing functions; AMR Speech CODEC; General description".

[19] (void)

[20] 3GPP TS 26.171: "AMR Wideband Speech Codec; General Description".

[21] ISO/IEC 14496-3:2005: "Information technology – Coding of audio-visual objects – Part 3: Audio".

[22] ITU-T Recommendation H.263 (01/05): "Video coding for low bit rate communication".

[23] (void)

[24] ISO/IEC 14496-2:2004: "Information technology – Coding of audio-visual objects – Part 2: Visual".

[25] (void)

[26] ITU-T Recommendation T.81 (1992) | ISO/IEC 10918-1:1993: "Information technology – Digital compression and coding of continuous-tone still images – Requirements and guidelines".

[27] C-Cube Microsystems: "JPEG File Interchange Format", Version 1.02, September 1, 1992.

[28] W3C Recommendation: "XHTML Basic", http://www.w3.org/TR/2000/REC-xhtml-basic-20001219, December 2000.

[29] ISO/IEC 10646-1:2000: "Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane".

[30] IETF RFC 3629: 'UTF-8, a transformation format of ISO 10646', F. Yergeau, November 2003.

[31] W3C Recommendation: "Synchronized Multimedia Integration Language (SMIL 2.0)-[Second Edition]", http://www.w3.org/TR/2005/REC-SMIL2-20050107/, January 2005.

[32] CompuServe Incorporated: "GIF Graphics Interchange Format: A Standard defining a mechanism for the storage and transmission of raster-based graphics information", Columbus, OH, USA, 1987.

[33] CompuServe Incorporated: "Graphics Interchange Format: Version 89a", Columbus, OH, USA, 1990.

[34] (void)

[35] (void)

[36] (void)

[37] (void)

[38] IETF RFC 2083: "PNG (Portable Networks Graphics) Specification Version 1.0", Boutell T., et al., March 1997.

[39] W3C Recommendation: "Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0", http://www.w3.org/TR/2004/REC-CCPP-struct-vocab-20040115/, January 2004.

[40] Open Mobile Alliance: "User Agent Profile Version 2.0", February 2006.

[41] W3C Recommendation: "RDF Vocabulary Description Language 1.0: RDF Schema", http://www.w3.org/TR/2004/REC-rdf-schema-20040210/, February 2004.

[42] W3C Last Call Working Draft: "Scalable Vector Graphics (SVG) 1.2", http://www.w3.org/TR/2004/WD-SVG12-20041027/, October 2004.

[43] W3C Last Call Working Draft: "Mobile SVG Profile: SVG Tiny, Version 1.2", http://www.w3.org/TR/2004/WD-SVGMobile12-20040813/, August 2004.

[44] Scalable Polyphony MIDI Specification Version 1.0, RP-34, MIDI Manufacturers Association, Los Angeles, CA, February 2002.

[45] Scalable Polyphony MIDI Device 5-to-24 Note Profile for 3GPP Version 1.0, RP-35, MIDI Manufacturers Association, Los Angeles, CA, February 2002.

[46] "Standard MIDI Files 1.0", RP-001, in "The Complete MIDI 1.0 Detailed Specification, Document Version 96.1", The MIDI Manufacturers Association, Los Angeles, CA, USA, February 1996.

[47] WAP Forum Specification: "XHTML Mobile Profile", http://www1.wapforum.org/tech/terms.asp?doc=WAP-277-XHTMLMP-20011029-a.pdf, October 2001.

[48] (void)

[49] (void)

[50] 3GPP TS 26.244: "Transparent end-to-end packet switched streaming service (PSS); 3GPP file format (3GP)".

[51] 3GPP TS 26.245: "Transparent end-to-end packet switched streaming service (PSS); Timed text format".

[52] 3GPP TS 26.246: "Transparent end-to-end packet switched streaming service (PSS); 3GPP SMIL Language Profile".

[53] IETF RFC 4234: "Augmented BNF for Syntax Specifications: ABNF", Crocker D. and Overell P., October 2005.

[54] IETF RFC 3066: "Tags for Identification of Languages", Alvestrand H., January 2001.

[55] IETF RFC 3556: "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", Casner S., July 2003.

[56] 3GPP TS 23.107: "Quality of Service (QoS) concept and architecture".

[57] IETF RFC 4585: "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", Ott J. et al., July 2006.

[58] IETF RFC 3611: "RTP Control Protocol Extended Reports (RTCP XR)", Friedman T., Caceres R. and Clark A., November 2003.

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

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

[61] (void)

[62] (void)

[63] 3GPP TS 26.090: "Mandatory Speech Codec speech processing functions; Adaptive Multi-Rate (AMR) speech codec; Transcoding functions".

[64] 3GPP TS 26.073: "ANSI-C code for the Adaptive Multi Rate (AMR) speech codec".

[65] 3GPP TS 26.104: "ANSI-C code for the floating-point Adaptive Multi Rate (AMR) speech codec".

[66] 3GPP TS 26.190: "Speech Codec speech processing functions; AMR Wideband speech codec; Transcoding functions".

[67] 3GPP TS 26.173: "ANCI-C code for the Adaptive Multi Rate - Wideband (AMR-WB) speech codec".

[68] 3GPP TS 26.204: "ANSI-C code for the Floating-point Adaptive Multi-Rate Wideband (AMR-WB) speech codec".

[69] IETF RFC 4648: "The Base16, Base32, and Base64 Data Encodings", Josefsson S., October 2006.

[70] Mobile DLS, MMA specification v1.0. RP-41 Los Angeles, CA, USA. 2004.

[71] Mobile XMF Content Format Specification, MMA specification v1.0., RP-42, Los Angeles, CA, USA. 2004.

[72] IETF RFC 3711: "The Secure Real-time Transport Protocol (SRTP)", Baugher M. et al, March 2004.

[73] Bellovin, S., "Problem Areas for the IP Security Protocols" in Proceedings of the Sixth Usenix Unix Security Symposium, pp. 1-16, San Jose, CA, July 1996

[74] Open Mobile Alliance: "DRM Specification 2.0".

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

[76] IETF RFC 3675: "IPv6 Jumbograms", Borman D., Deering S. and Hinden R., August 1999.

[77] NIST, "Advanced Encryption Standard (AES)", FIPS PUB 197, http://www.nist.gov/aes/.

[78] IETF RFC 3394: "Advanced Encryption Standard (AES) Key Wrap Algorithm", Schaad J. and Housley R., September 2002.

[79] IETF RFC 3839: "MIME Type Registrations for 3rd Generation Partnership Project (3GPP) Multimedia files", Castagno R. and Singer D., July 2004.

[80] IETF RFC 4396: "RTP Payload Format for 3rd Generation Partnership Project (3GPP) Timed Text", Rey J. and Matsui Y., February 2006.

[81] IETF RFC 4588: "RTP Retransmission Payload Format", Rey J. et al., July 2006.

[82] 3GPP TS 26.290: "Extended AMR Wideband codec; Transcoding functions".

[83] 3GPP TS 26.304: "ANSI-C code for the Floating-point; Extended AMR Wideband codec".

[84] 3GPP TS 26.273: "ANSI-C code for the Fixed-point; Extended AMR Wideband codec".

[85] IETF RFC 4352: "RTP Payload Format for the Extended Adaptive Multi-Rate Wideband (AMR-WB+) Audio Codec", Sjoberg J. et al., January 2006.

[86] 3GPP TS 26.401: "General audio codec audio processing functions; Enhanced aacPlus general audio codec; General description".

[87] 3GPP TS 26.410: "General audio codec audio processing functions; Enhanced aacPlus general audio codec; Floating-point ANSI-C code".

[88] 3GPP TS 26.411: "General audio codec audio processing functions; Enhanced aacPlus general audio codec; Fixed-point ANSI-C code".

[89] (void)

[90] ITU-T Recommendation H.264 (03/09): "Advanced video coding for generic audiovisual services" | ISO/IEC 14496-10:2009: "Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding".

[91] (void)

[92] IETF RFC 3984: "RTP Payload Format for H.264 Video", Wenger S. et al, February 2005.

[93] IETF RFC 3890: "A Transport Independent Bandwidth Modifier for the Session Description Protocol (SDP)", Westerlund M., September 2004.

[94] Standard ECMA-327: "ECMAScript 3rd Edition Compact Profile", June 2001.

[95] 3GPP TR 26.936: "Performance characterization of 3GPP audio codecs".

[96] IETF RFC 4288: "Media Type Specifications and Registration Procedures", Freed N. and Klensin J., December 2005.

[97] IETF RFC 4613: "Media Type Registrations for Downloadable Sounds for Musical Instrument Digital Interface (MIDI)", Frojdh P., Lindgren U. and Westerlund M., September 2006.

[98] 3GPP TS 26.142: "Dynamic and Interactive Multimedia Scene".

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

[100] IETF RFC 1123: "Requirements for Internet Hosts -- Application and Support", Braden R., October 1989.

[101] Internet Streaming Media Alliance (ISMA), ISMA Encryption and Authentication, Version 1.1 release version, September 2006.

[102] Internet Streaming Media Alliance (ISMA), ISMA Encryption and Authentication, Version 2.0 release version, November 2007.

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

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

[105] RFC 2818: HTTP Over TLS, E. Rescorla, May 2000.

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

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

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

[109] 3GPP TS 26.430: "Timed Graphics".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

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

discrete media: media that itself does not contain an element of time. In the present document all media not defined as continuous media

device capability description: a description of device capabilities and/or user preferences. Contains a number of capability attributes

device capability profile: same as device capability description

kilobits: 1000 bits

kilobytes: 1024 bytes

presentation description: contains information about one or more media streams within a presentation, such as the set of encodings, network addresses and information about the content

PSS client: client for the 3GPP packet switched streaming service based on the IETF RTSP/SDP and/or HTTP standards, with possible additional 3GPP requirements according to the present document

PSS server: server for the 3GPP packet switched streaming service based on the IETF RTSP/SDP and/or HTTP standards, with possible additional 3GPP requirements according to the present document

scene description: description of the spatial layout and temporal behaviour of a presentation. It can also contain hyperlinks

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in 3GPP TR 21.905 [3] and the following apply.

3GP	3GPP file format
AAC	Advanced Audio Coding
ADU	Application Data Unit
AVC	Advanced Video Coding
CC/PP	Composite Capability / Preference Profiles
DCT	Discrete Cosine Transform
DIMS	Dynamic and Interactive Multimedia Scenes
DLS	Downloadable Sounds
DRM	Digital Rights Management
Enhanced aacPlus	MPEG-4 High Efficiency AAC plus MPEG-4 Parametric Stereo
GIF	Graphics Interchange Format
HTML	Hyper Text Markup Language
ITU-T	International Telecommunications Union – Telecommunications
JFIF	JPEG File Interchange Format
MIDI	Musical Instrument Digital Interface
MIME	Multipurpose Internet Mail Extensions
MMS	Multimedia Messaging Service
NADU	Next Application Data Unit
PNG	Portable Networks Graphics
PSS	Packet-switched Streaming Service
QCIF	Quarter Common Intermediate Format
RDF	Resource Description Framework
RTCP	RTP Control Protocol
RTP	Real-time Transport Protocol
RTSP	Real-Time Streaming Protocol
SBR	Spectral Band Replication
SDP	Session Description Protocol
SMIL	Synchronised Multimedia Integration Language
SP-MIDI	Scalable Polyphony MIDI
SRTP	The Secure Real-time Transport Protocol
SVG	Scalable Vector Graphics
UAProf	User Agent Profile
UCS-2	Universal Character Set (the two octet form)
UTF-8	Unicode Transformation Format (the 8-bit form)
W3C	WWW Consortium
WML	Wireless Markup Language
XHTML	eXtensible Hyper Text Markup Language
XMF	eXtensible Music Format
XML	eXtensible Markup Language

4 System description

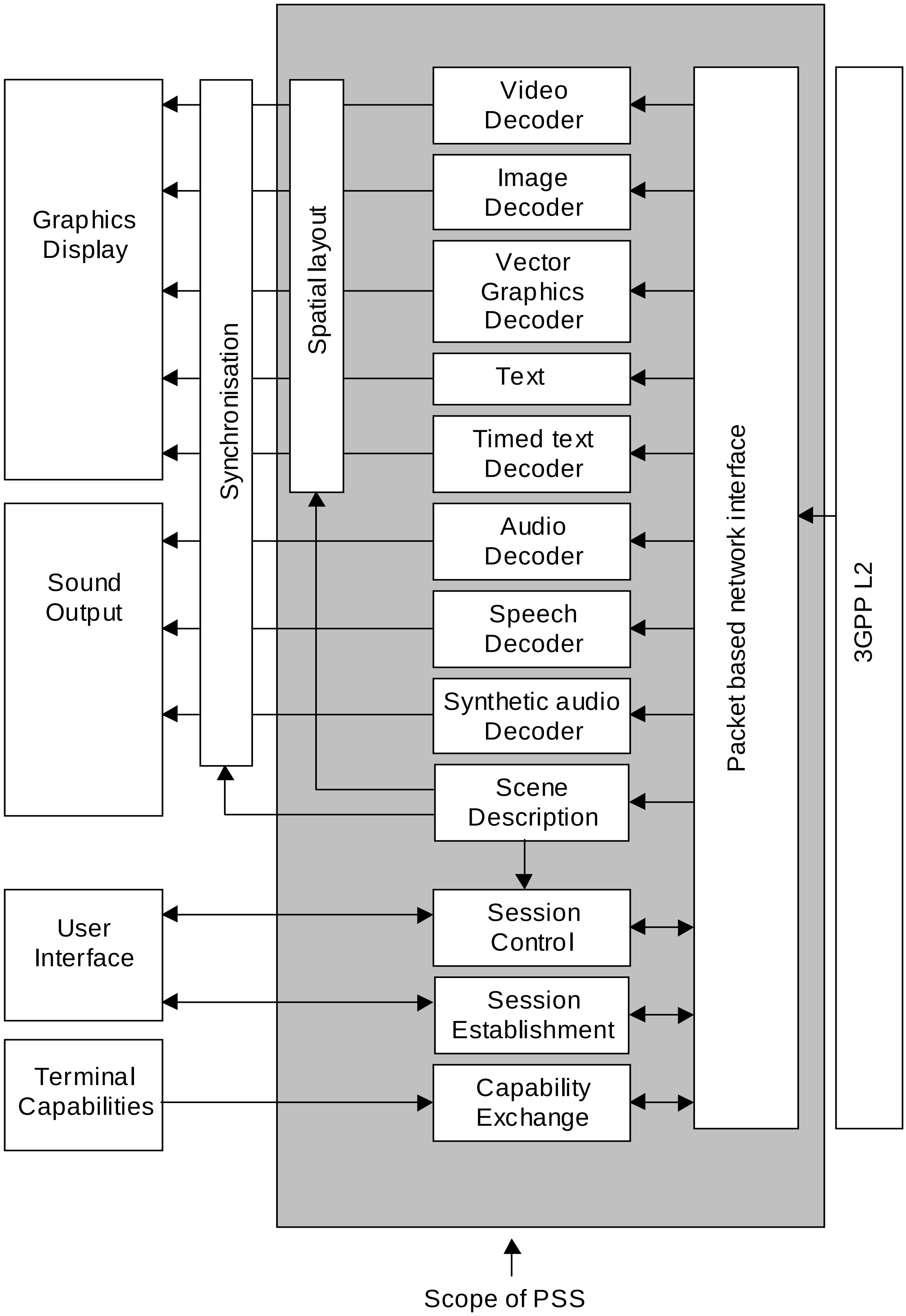


Figure 1: Functional components of a PSS client

Figure 1 shows the functional components of a PSS client. Figure 2 gives an overview of the protocol stack used in a PSS client and also shows a more detailed view of the packet based network interface. The functional components can be divided into control, scene description, media codecs and the transport of media and control data.

The control related elements are session establishment, capability exchange and session control (see clause 5).

- Session establishment refers to methods to invoke a PSS session from a browser or directly by entering an URL in the terminal's user interface.
- Capability exchange enables choice or adaptation of media streams depending on different terminal capabilities.
- Session control deals with the set-up of the individual media streams between a PSS client and one or several PSS servers. It also enables control of the individual media streams by the user. It may involve VCR-like presentation control functions like start, pause, fast forward and stop of a media presentation.

The scene description consists of spatial layout and a description of the temporal relation between different media that is included in the media presentation. The first gives the layout of different media components on the screen and the latter controls the synchronisation of the different media (see clause 8).

The PSS includes media codecs for video, still images, vector graphics, bitmap graphics, text, timed text, natural and synthetic audio, and speech (see clause 7).

Transport of media and control data consists of the encapsulation of the coded media and control data in a transport protocol (see clause 6). This is shown in figure 1 as the "packet based network interface" and displayed in more detail in the protocol stack of figure 2.

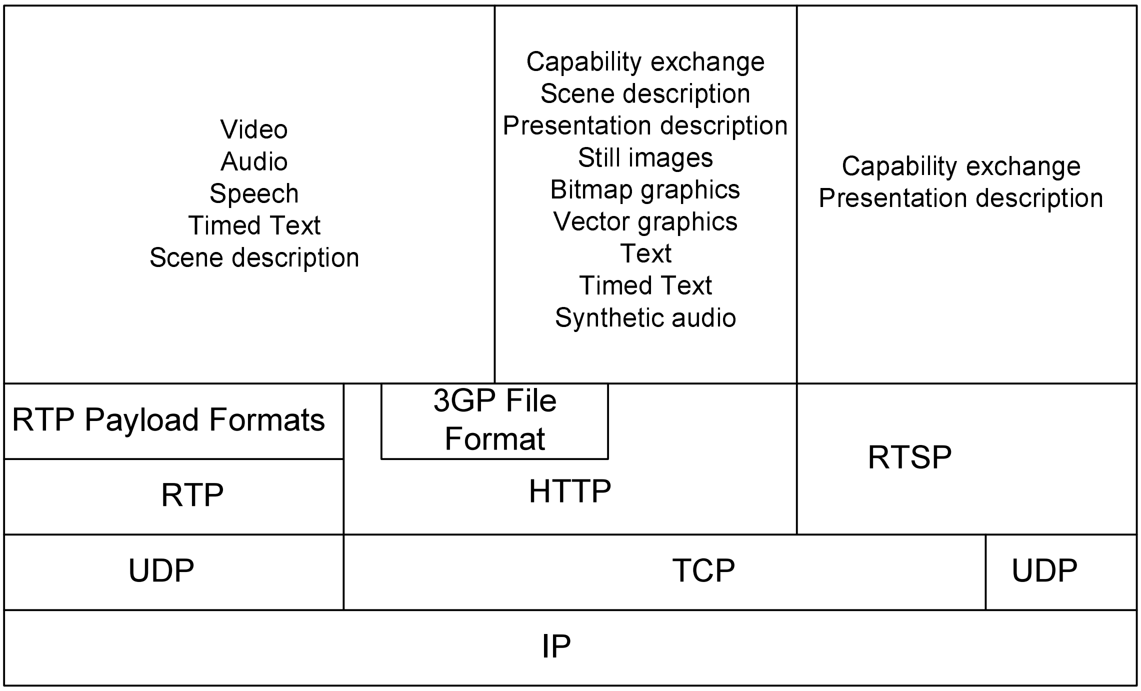


Figure 2: Overview of the protocol stack

The delivery of media over HTTP provides an alternative delivery mechanism to the RTSP/RTP based media delivery. HTTP Progressive download is described in clause 5.1. Adaptive HTTP streaming is described in clause 12.

5 Protocols

5.1 Session establishment

Session establishment refers to the method by which a PSS client obtains the initial session description. The initial session description can e.g. be a presentation description, a scene description or just an URL to the content.

A PSS client shall support initial session descriptions specified in one of the following formats: SMIL, SDP, or plain RTSP URL.

In addition to rtsp:// the PSS client shall support URLs [60] to valid initial session descriptions starting with file:// (for locally stored files) and http:// (for presentation descriptions or scene descriptions delivered via HTTP).

Examples for valid inputs to a PSS client are: file://temp/morning_news.smil, http://example.com/morning_news.sdp, and rtsp://example.com/morning_news.

URLs can be made available to a PSS client in many different ways. It is out of the scope of this specification to mandate any specific mechanism. However, an application using the 3GPP PSS shall at least support URLs of the above type, specified or selected by the user.

The preferred way would be to embed URLs to initial session descriptions within HTML or WML pages. Browser applications that support the HTTP protocol could then download the initial session description and pass the content to the PSS client for further processing. How exactly this is done is an implementation specific issue and out of the scope of this specification.

As an alternative to conventional streaming, a PSS client should also support progressive download of 3GP files [50] delivered via HTTP. A progressive-download session is established with one or more HTTP GET requests. In order to improve playback performance for 3GP files that are not authored for progressive download, a PSS client may issue (multiple pipelined) HTTP GET requests with byte ranges [17]. Example of a valid URL is http://example.com/morning_news.3gp.

5.2 Capability exchange

5.2.1 General

Capability exchange is an important functionality in the PSS. It enables PSS servers to provide a wide range of devices with content suitable for the particular device in question. Another very important task is to provide a smooth transition between different releases of PSS. Therefore, PSS clients and servers should support capability exchange.

The specification of capability exchange for PSS is divided into two parts. The normative part contained in clause 5.2 and an informative part in clause A.4 in Annex A of the present document. The normative part gives all the necessary requirements that a client or server shall conform to when implementing capability exchange in the PSS. The informative part provides additional important information for understanding the concept and usage of the functionality. It is recommended to read clause A.4 in Annex A before continuing with clauses 5.2.2-5.2.7.

5.2.2 The device capability profile structure

A device capability profile is an RDF [41] document that follows the structure of the CC/PP framework [39] and the CC/PP application UAProf [40]. Attributes are used to specify device capabilities and preferences. A set of attribute names, permissible values and semantics constitute a CC/PP vocabulary, which is defined by an RDF schema. For PSS, the UAProf vocabulary is reused and an additional PSS specific vocabulary is defined. The details can be found in clause 5.2.3. The syntax of the attributes is defined in the vocabulary schema, but also, to some extent, the semantics. A PSS device capability profile is an instance of the schema (UAProf and/or the PSS specific schema) and shall follow the rules governing the formation of a profile given in the CC/PP specification [39]. The profile schema shall also be governed by the rules defined in UAProf [40] chapter 7, 7.1, 7.3 and 7.4.

5.2.3 Vocabularies for PSS

5.2.3.1 General

Clause 5.2.3 specifies the attribute vocabularies to be used by the PSS capability exchange.

PSS servers supporting capability exchange shall support the attributes in the four PSS components of the PSS base vocabulary. PSS servers should also support the recommended attributes from the UAProf vocabulary [40]. A server may additionally support other UAProf attributes.

5.2.3.2 PSS base vocabulary

The PSS base vocabulary contains four components called "PssCommon", "Streaming", "ThreeGPFileFormat" and "PssSmil". The division of the vocabulary into these components is motivated by the fact that the PSS contains three different base applications:

- pure RTSP/RTP-based streaming (described by the Streaming component);
- 3GP file download or progressive download (described by the ThreeGPFileFormat component);
- SMIL presentation (described by the PssSmil component).

The last application can consist of downloadable images, text, etc., as well as RTSP/RTP streaming and downloadable 3GP files. Capabilities that are common to all PSS applications are described by the PssCommon component. The three base applications are distinguished from each other by the source of synchronization: for pure streaming it is RTP, for 3GP files it is inherit in the 3GP file format, and for SMIL presentations timing is provided by the SMIL file.

NOTE: DIMS presentations can be streamed (RTSP/RTP) or downloaded (as 3GP files). Capabilities for such presentations are described by the Streaming component and the ThreeGPFileFormat component, respectively. See in particular the StreamingAccept and ThreeGPAccept attributes below.

The PSS base vocabulary is an extension to UAProf and is defined as an RDF schema in Annex F. Together with the description of the attributes in the present clause, it defines the vocabulary. The vocabulary is associated with an XML namespace, which combines a base URI with a local XML element name to yield an URI. Annex F provides the details.

The PSS specific components contain a number of attributes expressing capabilities. The following subclauses list all attributes for each component.

5.2.3.2.1 PssCommon component

Attribute name: **AudioChannels**

Attribute definition: This attribute describes the stereophonic capability of the natural audio device.

Component: PssCommon

Type: Literal

Legal values: 'Mono', 'Stereo'

Resolution rule: Locked

EXAMPLE 1: `<AudioChannels>Mono</AudioChannels>`

Attribute name: **MaxPolyphony**

Attribute definition: The MaxPolyphony attribute refers to the maximal polyphony that the synthetic audio device supports as defined in [44].

NOTE: The MaxPolyphony attribute is 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 by setting the MaxPolyphony attribute the PSS client is required to support Scalable Polyphony MIDI i.e. Channel Masking defined in [44].

Component: PssCommon

Type: Number

Legal values: Integer between 5 and 24

Resolution rule: Locked

EXAMPLE 2: `<MaxPolyphony>8</MaxPolyphony>`

Attribute name: **NumOfGM1Voices**

Attribute definition: The NumOfGM1Voices attribute refers to the maximum number of simultaneous GM1 voices that the synthetic audio engine supports.

Component: PssCommon

Type: Number

Legal values: Integer greater or equal than 5

Resolution rule: Locked

EXAMPLE 3: `<NumOfGMIVoices>24</NumOfGMIVoices>`

Attribute name: **NumOfMobileDLSVoicesWithoutOptionalBlocks**

Attribute definition: The NumOfMobileDLSVoicesWithoutOptionalBlocks attribute refers to the maximum number of simultaneous Mobile DLS [70] voices without optional group of processing blocks that the synthetic audio engine supports.

Component: PssCommon

Type: Number

Legal values: Integer greater or equal than 5

Resolution rule: Locked

EXAMPLE 4:
```
<NumOfMobileDLSVoicesWithoutOptionalBlocks>24
</NumOfMobileDLSVoicesWithoutOptionalBlocks>
```

Attribute name: **NumOfMobileDLSVoicesWithOptionalBlocks**

Attribute definition: The NumOfMobileDLSVoicesWithOptionalBlocks attribute refers to the maximum number of simultaneous Mobile DLS 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.

Component: PssCommon

Type: Number

Legal values: Integer greater than or equal to 0

Resolution rule: Locked

EXAMPLE 5:
```
<NumOfMobileDLSVoicesWithOptionalBlocks>24
</NumOfMobileDLSVoicesWithOptionalBlocks>
```

Attribute name: **PssVersion**

Attribute definition: Latest PSS version supported by the client.

Component: PssCommon

Type: Literal

Legal values: "3GPP-R4", "3GPP-R5", "3GPP-R6", "3GPP-R7" and so forth.

Resolution rule: Locked

EXAMPLE 6:
```
<PssVersion>3GPP-R6</PssVersion>
```

Attribute name: **RenderingScreenSize**

Attribute definition: 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.

Component: PssCommon

Type: Dimension

Legal values: Two integer values equal or greater than zero. A value equal '0x0'means that there exists no possibility to render visual PSS presentations.

Resolution rule: Locked

EXAMPLE 7:
```
<RenderingScreenSize>70x15</RenderingScreenSize>
```

5.2.3.2.2 Streaming component

Attribute name: **StreamingAccept**

Attribute definition: 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.

Component: Streaming

Type: Literal (Bag)

Legal values: List of MIME types with related parameters.

Resolution rule: Append

EXAMPLE 1:
```
<StreamingAccept>
  <rdf:Bag>
    <rdf:li>audio/AMR-WB; octet-alignment=1</rdf:li>
    <rdf:li>video/H263-2000; profile=0; level=45</rdf:li>
  </rdf:Bag>
</StreamingAccept>
```

EXAMPLE 1b:
```
<StreamingAccept>
  <rdf:Bag>
    <rdf:li>audio/AMR-WB+</rdf:li>
    <rdf:li>video/H264; profile-level-id=42e00a</rdf:li>
    <rdf:li>video/richmedia+xml; Version-profile=10</rdf:li>
  </rdf:Bag>
</StreamingAccept>
```

Attribute name: **StreamingAccept-Subset**

Attribute definition: 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 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 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.

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

Component: Streaming

Type: Literal (Bag)

Legal values: No subsets defined.

Resolution rule: Append

Attribute name: **ThreeGPPLinkChar**

Attribute definition: Indicates whether the device supports the 3GPP-Link-Char header according to clause 10.2.1.1.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Override

EXAMPLE 2: `<ThreeGPPLinkChar>Yes</ThreeGPPLinkChar>`

Attribute name: **AdaptationSupport**

Attribute definition: Indicates whether the device supports client buffer feedback signaling according to clause 10.2.3.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Locked

EXAMPLE 3: `<AdaptationSupport>Yes</AdaptationSupport>`

Attribute name: **QoESupport**

Attribute definition: Indicates whether the device supports QoE signaling according to clauses 5.3.2.3, 5.3.3.6, and 11.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Locked

EXAMPLE 3: `<QoESupport>Yes</QoESupport>`

Attribute name: **ExtendedRtcpReports**

Attribute definition: Indicates whether the device supports extended RTCP reports according to clause 6.2.3.1.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Locked

EXAMPLE 4: `<ExtendedRtcpReports>Yes</ExtendedRtcpReports>`

Attribute name: **RtpRetransmission**

Attribute definition: Indicates whether the device supports RTP retransmission according to clause 6.2.3.3.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Locked

EXAMPLE 5: `<RtpRetransmission>Yes</RtpRetransmission>`

Attribute name: **MediaAlternatives**

Attribute definition: 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.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Override

EXAMPLE 6: `<MediaAlternatives>Yes</MediaAlternatives>`

Attribute name: **RtpProfiles**

Attribute definition: List of supported RTP profiles.

Component: Streaming

Type: Literal (Bag)

Legal values: Profile names registered through the Internet Assigned Numbers Authority (IANA), www.iana.org.

Resolution rule: Append

EXAMPLE 7:
```
<RtpProfiles>
  <rdf:Bag>
    <rdf:li>RTP/AVP</rdf:li>
    <rdf:li>RTP/AVPF</rdf:li>
  </rdf:Bag>
</RtpProfiles>
```

Attribute name: **StreamingOmaDrm**

Attribute definition: Indicates whether the device supports streamed OMA DRM protected content as defined by OMA and Annex K.

Component: Streaming

Type: Literal (Bag)

Legal values: OMA Version numbers supported as a floating number. 0.0 indicates no support.

Resolution rule: Locked

EXAMPLE 8:
```
<StreamingOmaDrm>
  <rdf:Bag>
    <rdf:li>2.0</rdf:li>
  </rdf:Bag>
</StreamingOmaDrm>
```

Attribute name: **PSSIntegrity**

Attribute definition: Indicates whether the device supports integrity protection for streamed content as defined by Annex K.2.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Locked

EXAMPLE 9:
```
<PSSIntegrity>Yes</PSSIntegrity>
```

Attribute name: **VideoDecodingByteRate**

Attribute definition: 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).

Component: Streaming

Type: Number

Legal values: Integer value greater than or equal to 16000.

Resolution rule: Locked

EXAMPLE 10:
```
<VideoDecodingByteRate>16000</VideoDecodingByteRate>
```

Attribute name: **VideoInitialPostDecoderBufferingPeriod**

Attribute definition: 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-decoder buffering.

Component: Streaming

Type: Number

Legal values: Integer value equal to or greater than zero.

Resolution rule: Locked

EXAMPLE 11:
```
<VideoInitialPostDecoderBufferingPeriod>9000
</VideoInitialPostDecoderBufferingPeriod>
```

Attribute name: **VideoPreDecoderBufferSize**

Attribute definition: 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.

Component: Streaming

Type: Number

Legal values: Integer value equal to or greater than zero. Values greater than one but less than the default buffer size defined in Annex G are not allowed.

Resolution rule: Locked

EXAMPLE 12:
```
<VideoPreDecoderBufferSize>30720</VideoPreDecoderBufferSize>
```

5.2.3.2.3 ThreeGPFileFormat component

Attribute name: **Brands**

Attribute definition: List of supported 3GP profiles identified by brand.

Component: ThreeGPFileFormat

Type: Literal (Bag)

Legal values: Brand identifiers according to 5.3.4 and 5.4 in [50].

Resolution rule: Append

EXAMPLE 1:
```
<Brands>
  <rdf:Bag>
    <rdf:li>3gp4</rdf:li>
    <rdf:li>3gp5</rdf:li>
    <rdf:li>3gp6</rdf:li>
    <rdf:li>3gr6</rdf:li>
    <rfd:li>3gp7</rdf:li>
    <rfd:li>3gr7</rdf:li>
    <rfd:li>3ge7</rdf:li>
  </rdf:Bag>
</Brands>
```

Attribute name: **ThreeGPAccept**

Attribute definition: List of content types (MIME types) that can be included in a 3GP file and handled by the PSS application. The content types included in this attribute can be rendered in a 3GP file or a presentation contained therein. If the identifier "Streaming-Media" is included, streaming media can be included within a contained 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.

Component: ThreeGPFileFormat

Type: Literal (Bag)

Legal values: List of MIME types with related parameters and the "Streaming-Media" identifier.

Resolution rule: Append

EXAMPLE 2:
```
<ThreeGPAccept>
  <rdf:Bag>
    <rdf:li>video/H263-2000; profile=0; level=45</rdf:li>
    <rdf:li>audio/AMR</rdf:li>
  </rdf:Bag>
</ThreeGPAccept>
```

EXAMPLE 2b:
```
<ThreeGPAccept>
  <rdf:Bag>
    <rdf:li>audio/AMR</rdf:li>
    <rdf:li>audio/AMR-WB+</rdf:li>
    <rdf:li>video/H263-2000; profile=0; level=45</rdf:li>
    <rdf:li>video/H264; profile-level-id=42e00a</rdf:li>
    <rdf:li>image/jpeg</rdf:li>
    <rdf:li>video/richmedia+xml; Version-profile=10</rdf:li>
    <rdf:li>Streaming-Media</rdf:li>
  </rdf:Bag>
</ThreeGPAccept>
```

Attribute name: **ThreeGPAccept-Subset**

Attribute definition: 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.

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

Component: ThreeGPFileFormat

Type: Literal (Bag)

Legal values: No subsets defined.

Resolution rule: Append

Attribute name: **ThreeGPOmaDrm**

Attribute definition: List of the OMA DRM versions that is supported to be used for DRM protection of content present in the 3GP file format.

Component: ThreeGPFileFormat

Type: Literal (Bag)

Legal values: OMA DRM version numbers as floating point values. 0.0 indicates no support.

Resolution rule: Locked

EXAMPLE 3:
```
<3gpOMADRM>
  <rdf:Bag>
    <rdf:li>2.0 </rdf:li>
```

```
  </rdf:Bag>
</3gpOMADRM>
```

5.2.3.2.4 PssSmil component

Attribute name: **SmilAccept**

Attribute definition: 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.

Component: PssSmil

Type: Literal (Bag)

Legal values: List of MIME types with related parameters and the "Streaming-Media" identifier.

Resolution rule: Append

EXAMPLE 1:

```
<SmilAccept>
  <rdf:Bag>
    <rdf:li>image/gif</rdf:li>
    <rdf:li>image/jpeg</rdf:li>
    <rdf:li>Streaming-Media</rdf:li>
  </rdf:Bag>
</SmilAccept>
```

Attribute name: **SmilAccept-Subset**

Attribute definition: 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 present document are supported.
- "SVG-Tiny"
- "SVG-Basic"

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

Component: PssSmil

Type: Literal (Bag)

Legal values: "JPEG-PSS", "SVG-Tiny", "SVG-Basic"

Resolution rule: Append

EXAMPLE 2:
```
<SmilAccept-Subset>
  <rdf:Bag>
    <rdf:li>JPEG-PSS</rdf:li>
    <rdf:li>SVG-Tiny</rdf:li>
  </rdf:Bag>
</SmilAccept-Subset>
```

Attribute name: **SmilBaseSet**

Attribute definition: Indicates a base set of SMIL 2.0 modules that the client supports.

Component: Streaming

Type: Literal

Legal values: 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 present document (TS 26.234) and of TS 26.246 [52].

Resolution rule: Locked

EXAMPLE 3:
```
<SmilBaseSet>SMIL-3GPP-R6</SmilBaseSet>
```

Attribute name: **SmilModules**

Attribute definition: 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.

Component: Streaming

Type: Literal (Bag)

Legal values: SMIL 2.0 module names defined in the SMIL 2.0 recommendation [31], section 2.3.3, table 2.

Resolution rule: Append

EXAMPLE 4:
```
<SmilModules>
  <rdf:Bag>
    <rdf:li>BasicTransitions</rdf:li>
    <rdf:li>MulitArcTiming</rdf:li>
  </rdf:Bag>
</SmilModules>
```

5.2.3.3 Attributes from UAProf

In the UAProf vocabulary [40] there are several attributes that are of interest for the PSS. The formal definition of these attributes is given in [40]. The following list of attributes is recommended for PSS applications:

Attribute name: **BitsPerPixel**

Component: HardwarePlatform

Attribute description: The number of bits of colour or greyscale information per pixel

EXAMPLE 1:
```
<BitsPerPixel>8</BitsPerPixel>
```

Attribute name: **ColorCapable**

Component: HardwarePlatform

Attribute description: Whether the device display supports colour or not.

EXAMPLE 2:
```
<ColorCapable>Yes</ColorCapable>
```

Attribute name: **PixelAspectRatio**

Component: HardwarePlatform

Attribute description: Ratio of pixel width to pixel height

EXAMPLE 3:
```
<PixelAspectRatio>1x2</PixelAspectRatio>
```

Attribute name: **PointingResolution**

Component: HardwarePlatform

Attribute description: Type of resolution of the pointing accessory supported by the device.

EXAMPLE 4:
```
<PointingResolution>Pixel</PointingResolution>
```

Attribute name: **Model**

Component: HardwarePlatform

Attribute description: Model number assigned to the terminal device by the vendor or manufacturer

EXAMPLE 5:
```
<Model>Model B</Model>
```

Attribute name: **Vendor**

Component: HardwarePlatform

Attribute description: Name of the vendor manufacturing the terminal device

EXAMPLE 6:
```
<Vendor>TerminalManufacturer A</Vendor>
```

Attribute name: **CcppAccept-Charset**

Component: SoftwarePlatform

Attribute description: List of character sets the device supports

EXAMPLE 7:
```
<CcppAccept-Charset>
  <rdf:Bag>
    <rdf:li>UTF-8</rdf:li>
  </rdf:Bag>
</CcppAccept-Charset>
```

Attribute name: **CcppAccept-Encoding**

Component: SoftwarePlatform

Attribute description: List of transfer encodings the device supports

EXAMPLE 8:
```
<CcppAccept-Encoding>
  <rdf:Bag>
    <rdf:li>base64</rdf:li>
  </rdf:Bag>
</CcppAccept-Encoding>
```

Attribute name: **CcppAccept-Language**

Component: SoftwarePlatform

Attribute description: List of preferred document languages

EXAMPLE 9:
```
<CcppAccept-Language>
  <rdf:Seq>
    <rdf:li>en</rdf:li>
    <rdf:li>se</rdf:li>
  </rdf:Seq>
</CcppAccept-Language>
```

5.2.4 Extensions to the PSS schema/vocabulary

5.2.4.1 Vocabulary definitions

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 Release-6 PSS profile schema specification has been updated from Release 5 and has thus been assigned a unique RDF schema. The same is true for the Release-7 PSS profile schema specification. The following URIs uniquely identify the RDF schemas for Release 5, Release 6 and Release 7:

PSS Release 5 URI: http://www.3gpp.org/profiles/PSS/ccppschema-PSS5#

PSS Release 6 URI: http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#

PSS Release 7 URI: http://www.3gpp.org/profiles/PSS/ccppschema-PSS7#

In the future new usage scenarios might have need for expressing new attributes. If the base vocabulary is further updated, a new unique namespace will be assigned to the updated schema. The base vocabulary shall only be changed by the TSG responsible for the present document. All extensions to the profile schema shall be governed by the rules defined in [40] clause 7.7.

5.2.4.2 Backward compatibility

An important issue when introducing a new vocabulary is to ensure backward compatibility. PSS Release-6 clients should seamlessly work together with PSS Release-5 servers and vice versa. To obtain backward compatibility, a Release-6 client should provide servers with multiple device-capability profiles using PSS Release-5 and Release-6 vocabularies, respectively. This can be done by providing two URIs referring to two separate profiles or one URI referring to one combined profile that uses both the Relase-5 and the Release-6 namespaces. PSS Release-6 servers should handle both namespaces, whereas PSS Release-5 servers will ignore profiles with unknown namespaces.

5.2.5 Signalling of profile information between client and server

When a PSS client or server support capability exchange it shall support the profile information transport over both HTTP and RTSP between client and server as defined in clause 9.1 (including its subsections) of the WAP 2.0 UAProf specification [40] with the following amendments:

- The "x-wap-profile" and "x-wap-profile-diff" headers shall be present in at least one HTTP or RTSP request per session. That is, the requirement to send this header in all requests has been relaxed.
- The defined headers may be applied to both RTSP and HTTP.
- The "x-wap-profile-diff" header is only valid for the current request. The reason is that PSS does not have the WSP session concept of WAP.
- Push is not relevant for the PSS.

The following guidelines concern how and when profile information is sent between client and server:

- PSS content servers supporting capability exchange shall be able to receive profile information in all HTTP and RTSP requests.
- The terminal should not send the "x-wap-profile-diff" header over the air-interface since there is no compression scheme defined.
- RTSP: the client should send profile information in the DESCRIBE message. It may send it in any other request.

If the terminal has some prior knowledge about the file type it is about to retrieve, e.g. file extensions, the following apply:

- HTTP and SDP: when retrieving an SDP with HTTP the client should include profile information in the GET request. This way the HTTP server can deliver an optimised SDP to the client.
- HTTP and SMIL: When retrieving a SMIL file with HTTP the client should include profile information in the GET request. This way the HTTP server can deliver an optimised SMIL presentation to the client. A SMIL presentation can include links to static media. The server should optimise the SMIL file so that links to the referenced static media are adapted to the requesting client. When the "x-wap-profile-warning" indicates that content selection has been applied (201-203) the PSS client should assume that no more capability exchange has to be performed for the static media components. In this case it should not send any profile information when retrieving static media to be included in the SMIL presentation. This will minimise the HTTP header overhead.

5.2.6 Merging device capability profiles

Profiles need to be merged whenever the PSS server receives multiple device capability profiles. Multiple occurrences of attributes and default values make it necessary to resolve the profiles according to a resolution process.

The resolution process shall be the same as defined in UAProf [40] clause 6.4.1.

- Resolve all indirect references by retrieving URI references contained within the profile.
- Resolve each profile and profile-diff document by first applying attribute values contained in the default URI references and by second applying overriding attribute values contained within the category blocks of that profile or profile-diff.
- Determine the final value of the attributes by applying the resolved attribute values from each profile and profile-diff in order, with the attribute values determined by the resolution rules provided in the schema. Where no resolution rules are provided for a particular attribute in the schema, values provided in profiles or profile-diffs are assumed to override values provided in previous profiles or profile-diffs.

When several URLs are defined in the "x-wap-profile" header and there exists any attribute that occurs more than once in these profiles the rule is that the attribute value in the second URL overrides, or is overridden by, or is appended to the attribute value from the first URL (according to the resolution rule) and so forth. This is what is meant with "Determine the final value of the attributes by applying the resolved attribute values from each profile and profile-diff in order, with…" in the third bullet above. If the profile is completely or partly inaccessible or otherwise corrupted the server should still provide content to the client. The server is responsible for delivering content optimised for the client based on the received profile in a best effort manner.

NOTE: For the reasons explained in Annex A clause A.4.3 the usage of indirect references in profiles (using the CC/PP defaults element) is not recommended.

5.2.7 Profile transfer between the PSS server and the device profile server

The device capability profiles are stored on a device profile server and referenced with URLs. According to the profile resolution process in clause 5.2.6 of the present document, the PSS server ends up with a number of URLs referring to profiles and these shall be retrieved.

- The device profile server shall support HTTP 1.1 for the transfer of device capability profiles to the PSS server.

- If the PSS server supports capability exchange it shall support HTTP 1.1 for transfer of device capability profiles from the device profile server. A URL shall be used to identify a device capability profile.

- Normal content caching provisions as defined by HTTP apply.

5.3 Session set-up and control

5.3.1 General

Continuous media is media that has an intrinsic time line. Discrete media on the other hand does not itself contain an element of time. In this specification speech, audio, video, timed text and DIMS belong to the first category and still images and text to the latter one.

Streaming of continuous media using RTP/UDP/IP (see clause 6.2) requires a session control protocol to set-up and control the individual media streams. For the transport of discrete media (images and text), vector graphics, timed text and synthetic audio this specification adopts the use of HTTP/TCP/IP (see clause 6.3). In this case there is no need for a separate session set-up and control protocol since this is built into HTTP. This clause describes session set-up and control of continuous media.

5.3.2 RTSP

RTSP [5] shall be used for session set-up and session control. PSS clients and servers shall follow the rules for minimal on-demand playback RTSP implementations in appendix D of [5]. In addition to this:

- PSS servers and clients shall implement the DESCRIBE method (see clause 10.2 in [5]);
- PSS servers and clients shall implement the Range header field (see clause 12.29 in [5]);
- PSS servers shall include the Range header field in all PLAY responses;
- PSS servers and clients should implement the SET_PARAMETER method (see clause 10.9 in [5]);
- PSS servers and clients should implement the Bandwidth header field (see clause 12.6 in [5].

Further additions to RTSP are specified in the following subclauses.

5.3.2.1 The 3GPP-Link-Char header

PSS servers and clients should implement the 3GPP-Link-Char header field.

To enable PSS clients to report the link characteristics of the radio interface to the PSS server, the "3GPP-Link-Char" RTSP header is defined. The header takes one or more arguments. The reported information should be taken from a QoS reservation (i.e. the QoS profile as defined in [56]). Note that this information is only valid for the wireless link and does not apply end-to-end. However, the parameters do provide constraints that can be used.

Three parameters are defined that can be included in the header, and future extensions are possible to define. Any unknown parameter shall be ignored. The three parameters are:

- "GBW": the link's guaranteed bit-rate in kilobits per second as defined by [56];
- "MBW": the link's maximum bit-rate in kilobits per second as defined by [56];
- "MTD": the link's maximum transfer delay, as defined by [56] in milliseconds.

The "3GPP-Link-Char" header syntax is defined below using ABNF [53]:

3gpplinkheader = "3GPP-Link-Char" ":" link-char-spec *("," 0*1SP link-char-spec) CRLF

link-char-spec = char-link-url *(";" 0*1SP link-parameters)

char-link-url = "url" "=" <">url<">

link-parameters = Guaranteed-BW / Max-BW / Max-Transfer-delay / extension-type

```
Guaranteed-BW       = "GBW" "=" 1*DIGIT  ; kbps
Max-BW              = "MBW" "=" 1*DIGIT ; kbps
Max-Transfer-delay  = "MTD" "=" 1*DIGIT  ; ms
extension-type      = token "=" (token / quoted-string)
DIGIT               = as defined in RFC 2326 [5]
token               = as defined in RFC 2326 [5]
quoted-string       = as defined in RFC 2326 [5]
url                 = as defined in RFC 2326 [5]
```

The "3GPP-Link-Char" header can be included in a request using any of the following RTSP methods: SETUP, PLAY, OPTIONS, and SET_PARAMETER. The header shall not be included in any response. The header can contain one or more characteristics specifications. Each specification contains a URI that can either be an absolute or a relative, any relative URI use the RTSP request URI as base. The URI points out the media component that the given parameters apply to. This can either be an individual media stream or a session aggregate.

If a QoS reservation (PDP context) is shared by several media components in a session the 3GPP-Link-Char header shall not be sent prior to the RTSP PLAY request. In this case the URI to use is the aggregated RTSP URI. If the QoS reservation is not shared (one PDP context per media) the media stream URI must be used in the 3GPP-Link-Char specification. If one QoS reservation (PDP context) per media component is used, the specification parameters shall be sent per media component.

The "3GPP-Link-Char" header should be included in a SETUP or PLAY request by the client, to give the initial values for the link characteristics. A SET_PARAMETER or OPTIONS request can be used to update the 3GPP-Link-Char values in a session currently playing. It is strongly recommended that SET_PARAMETER is used, as this has the correct semantics for the operation and also requires less overhead both in bandwidth and server processing. When performing updates of the parameters, all of the previous signalled values are undefined and only the given ones in the update are defined. This means that even if a parameter has not changed, it must be included in the update.

Example:

```
3GPP-LinkChar: url="rtsp://server.example.com/media.3gp"; GBW=32; MBW=128; MTD=2000
```

In the above example the header tells the server that its radio link has a QoS setting with a guaranteed bit-rate of 32 kbps, a maximum bit-rate of 128 kbps, and a maximum transfer delay of 2.0 seconds. These parameters are valid for the aggregate of all media components, as the URI is an aggregated RTSP URI.

5.3.2.2 The 3GPP-Adaptation header

PSS servers and clients should implement the 3GPP-Adaptation header field.

To enable PSS clients to set bit-rate adaptation parameters, a new RTSP request and response header is defined. The header can be used in the methods SETUP, PLAY, OPTIONS, and SET_PARAMETER. The header defined in ABNF [53] has the following syntax:

```
3GPP-adaptation-def = "3GPP-Adaptation" ":" adaptation-spec 0*("," adaptation-spec)
adaptation-spec     = url-def *adapt-params
adapt-params        = ";" buffer-size-def
                    / ";" target-time-def
url-def             = "url" "=" <"> url <">
buffer-size-def     = "size" "=" 1*9DIGIT ; bytes
target-time-def     = "target-time" "=" 1*9DIGIT; ms
url                 = ( absoluteURI / relativeURI )
```

absoluteURI and relativeURI are defined in RFC 3986 [60]. The base URI for any relative URI is the RTSP request URI.

The "3GPP-Adaptation" header shall be sent in responses to requests containing this header. The PSS server shall not change the values in the response header. The presence of the header in the response indicates to the client that the server acknowledges the request.

The buffer size signalled in the "3GPP-Adaptation" header shall correspond to reception, de-jittering, and, if used, de-interleaving buffer(s) that have this given amount of space for complete application data units (ADU), including the following RTP header and RTP payload header fields: RTP timestamp, and sequence numbers or decoding order numbers. The specified buffer size shall also include any Annex G pre-decoder buffer space used for this media, as the two buffers cannot be separated.

The target protection time signalled in the "target-time" parameter is the targeted minimum buffer level in milliseconds; that is, the minimum amount of playback time the client perceives necessary for interrupt-free playback. This value must be chosen such that the client is never in a buffering state if all media streams have reached or exceeded their target-time in buffered data and playout delay. Once this desired level of target protection is achieved, the server may utilize any additional resources to increase the quality of the media or to increase the buffer duration beyond that required by the target-time, or it may continue sending at the media rate in order to maintain a steady buffer state.

5.3.2.3 The Quality of Experience headers

5.3.2.3.1 Protocol initiation and termination

A new RTSP header is defined to enable the PSS client and server to negotiate which Quality of Experience (QoE) metrics the PSS client should send, how often they should be sent and how to turn the metrics transmission off. This header can be sent in requests and responses of RTSP methods SETUP, SET_PARAMETER, OPTIONS (with Session ID) and PLAY. The exact usage of this header is defined in clause 11. The header is defined in ABNF [53] as follows (see [53] for specifiers not defined here):

```
QoE-Header        = "3GPP-QoE-Metrics" ":" ("Off" / Measure-Spec *("," Measure-Spec)) CRLF

Measure-Spec      = Stream-URL";" ((Metrics ";" Sending-rate [";" Measure-Range]
                      [";" Measure-Resolution] *([";" Metrics-Server]) *([";" Parameter-Ext])) /   'Off')

Stream-URL        = "url" "="  <'>Rtsp-URL<'>

Metrics           = "metrics" "=" "{"Metrics-Name *("|" Metrics-Name) " }"

Metrics-Name      =  1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7e) ;VCHAR except ';', ',', '{' or '}'

Sending-Rate      = 'rate" "=" 1*DIGIT / "End"

Measure-Resolution  = "resolution" "=" 1*DIGIT ; in seconds

Metrics-Server    = "server" "=" "{" Server-Name *("|" Server-Name) "}"

Server-Name       = as defined in RFC 1123 [100]

Measure-Range     = "range" ":" Ranges-Specifier

Parameter-Ext     = 'On'/'Off'/ (1*DIGIT ['.' 1*DIGIT]) / (1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7c /
                    0x7e))

Ranges-Specifier = as defined in RFC 2326 [5]

Rtsp-URL          = as defined in RFC 2326 [5]
```

There are two ways to use this header:

- Using only the "Off" parameter is an indication that either server or client wants to cancel the metrics reporting.
- Using other parameters indicates a request to start the metrics transmission.

If "Stream-URL" is an RTSP Session Control URL, then "Metrics" applies to the RTSP session. If "Stream-URL" is an RTSP Media Control URL, then "Metrics" apply only to the indicated media component of the session.

QoE metrics with the same "Stream-URL", "Sending-rate" and "Measure-Range" shall be aggregated within a single "Measure-Spec" declaration. Otherwise, multiple "Stream-URL" declarations shall be used.

The "Metrics" field contains the list of names that describes the metrics/measurements that are required to be reported in a PSS session. The names that are not included in the "Metrics" field shall not be reported during the session.

The "Sending-Rate" shall be set, and it expresses the maximum time period in seconds between two successive QoE reports. If the "Sending-Rate" value is 0, then the client shall decide the sending time of the reports depending on the events occurred in the client. Values ≥ 1 indicate a precise reporting interval. The shortest interval is one second and the longest interval is undefined. The reporting interval can be different for different media, but it is recommended to maintain a degree of synchronization in order to avoid extra traffic in the uplink direction. The value "End" indicates that only one report is sent at the end of the session.

A default QoE reporting is done for each metric. The optional "Measure-Resolution" field, if present, indicates that XML QoE reporting shall be done instead. In this case the "Measure-Resolution" field splits the session duration into a number of equally sized periods where each period is of the length specified by the "Measure-Resolution" field. QoE metrics are calculated for each period and stored in the terminal, and all the stored metrics are then sent together according to the "Sending-Rate" field. This allows long reporting intervals (to save bandwidth) without losing good metric measurement resolution. It is recommended that the Sending-Rate is set to an integer multiple of the Measure-Resolution, or to "End".

Note that both "Sending-Rate" and "Measure-Resolution" shall be evaluated according to a real-time clock. This implies that the real-time intervals for measurements and reporting are not affected by changes in playback rate, for instance due to buffering.

The optional "Metrics-Server" field, if present, specifies that instead of the default RTSP reporting back to the streaming server, the QoE reports should be sent to a separate HTTP server. If more than one server is specified, the terminal shall randomly select one of them to be used during the session. The Metrics-Server parameter can only be used for XML reporting, that is, together with the Measure-Resolution parameter. The formatting of the HTTP reports is specified in sub-clause 5.3.2.3.3. If the PSS client does not support HTTP reporting it shall reject the "Metrics-Server" field during the QoE negotiation phase.

The optional "Measure-Range" field, if used, shall define the time range in the stream for which the QoE metrics will be reported. There shall be only one range per measurement specification. The range format shall be any of the formats allowed by the media. If the "Measure-Range" field is not present, the corresponding (media or session level) range attribute in SDP shall be used. If SDP information is not present, the metrics range shall be the whole session duration.

There shall be only one "3GPP-QoE-Metrics" header in one RTSP request or response.

5.3.2.3.2 Metrics feedback

The QoE metrics feedback can be conveyed in requests to the PSS server using the SET_PARAMETER, PAUSE or TEARDOWN methods by the "3GPP-QoE-Feedback" header. The header is defined in ABNF [53] as follows (see [53] for specifiers not defined here):

Feedbackheader = "3GPP-QoE-Feedback" ":" Feedback-Spec *("," Feedback-Spec) CRLF

Feedback-Spec = Stream-URL 1*(";" Parameters) [";" Measure-Range]

Stream-URL = as specified in clause 5.3.2.3.1

Parameters = Metrics-Name "=" "{" SP / (Measure *("|" Measure)) "}"

Metrics-Name = as defined in clause 5.3.2.3.1

Measure = Value [SP Timestamp]

Measure-Range = as defined in clause 5.3.2.3.1

Value = (["-"]1*DIGIT ["." *DIGIT]) / 1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7e) ;VCHAR except ';', ',', '{' or '}'

Timestamp = NPT-Time

NPT-Time = as defined in RFC 2326 [5]

"Stream-URL" is the RTSP session or media control URL that identifies the media the feedback parameter applies to.

The "Metrics-Name" field in the "Parameters" definition contains the name of the metrics/measurements and uses the same identifiers as the "3GPP-QoE-Metrics" header in clause 5.3.2.3.1.

The "Value" field indicates the results. There is the possibility that the same event occurs more than once during a monitoring period. In that case the metrics value may occur more than once indicating the number of events to the server. For the XML reporting format only one value is reported for each measurement resolution period.

The optional "Timestamp" (defined in NPT time) indicates the time when the event occurred or when the metric was calculated. If no events have occurred, it shall be reported with an empty set (only containing a space). The "Timestamp" feedback shall not be used for the XML reporting format.

The optional "Measure-Range" indicates the actual measurement period, for which this report is valid.

QoE metrics reporting should be done by the PSS client by using the SET_PARAMETER method. However, for more efficiency, RTSP PAUSE and TEARDOWN methods may also be used in particular cases, such as:

CASE 1: When sending the very last QoE report, the client should embed the QoE information into a TEARDOWN message.

CASE 2: When the client wants to pause the streaming flow, QoE information should be embedded into a PAUSE method. The PSS client should not send any QoE reports to the PSS server when the system is paused, since there is no media flow.

5.3.2.3.3 Metrics feedback over HTTP

If a specific metrics server has been configured the client should send QoE reports using the HTTP (RFC 2616 [73]) POST request carrying XML formatted metadata. Each QoE report is formatted in XML according the following XML schema. An informative example of a single reception report XML object is also given.

5.3.2.3.3.1 XML Syntax for a QoE Report

Below is the formal XML syntax of QoE report instances.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="urn:3gpp:metadata:2009:PSS:receptionreport"
xmlns="urn:3gpp:metadata:2009:PSS:receptionreport"
elementFormDefault="qualified">

<xs:element name="receptionReport" type="receptionReportType"/>

<xs:complexType name="receptionReportType">
    <xs:choice>
        <xs:element name="statisticalReport" type="starType"
            minOccurs="0" maxOccurs="unbounded"/>
        <xs:any namespace="##other" processContents="skip" minOccurs="0" maxOccurs="unbounded"/>
    </xs:choice>
</xs:complexType>

<xs:complexType name="starType">
    <xs:sequence>
        <xs:element name="fileURI" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="qoeMetrics" type="qoeMetricsType" minOccurs="0" maxOccurs="1"/>
        <xs:any namespace="##other" processContents="skip" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="serviceId" type="xs:string" use="optional"/>
    <xs:attribute name="clientId" type="xs:string" use="optional"/>
    <xs:attribute name="serviceURI" type="xs:anyURI" use="optional"/>
    <xs:anyAttribute processContents="skip"/>
</xs:complexType>

<xs:complexType name="qoeMetricsType">
    <xs:sequence>
        <xs:element name="medialevel_qoeMetrics" type="medialevel_qoeMetricsType"
```

```
            minOccurs="0" maxOccurs="unbounded"/>
        <xs:any namespace="##other" processContents="skip" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="totalRebufferingDuration" type="xs:doubleVectorType" use="optional"/>
    <xs:attribute name="numberOfRebufferingEvents" type="xs:unsignedLongVectorType"
        use="optional"/>
    <xs:attribute name="initialBufferingDuration" type="xs:double" use="optional"/>
    <xs:attribute name="contentSwitchTime" type="xs:doubleVectorType" use="optional"/>
    <xs:attribute name="sessionStartTime" type="xs:unsignedLong"/>
    <xs:attribute name="sessionStopTime" type="xs:unsignedLong"/>
    <xs:attribute name="bufferDepth" type="xs:doubleVectorType" use="optional"/>
    <xs:attribute name="allContentBuffered" type="xs:boolean" use="optional"/>
    <xs:anyAttribute processContents="skip"/>
</xs:complexType>

<xs:complexType name="medialevel_qoeMetricsType">
    <xs:attribute name="sessionId" type="xs:string"/>
    <xs:attribute name="totalCorruptionDuration" type="xs:unsignedLongVectorType" use="optional"/>
    <xs:attribute name="numberOfCorruptionEvents" type="xs:unsignedLongVectorType" use="optional"/>
    <xs:attribute name="t" type="xs:boolean" use="optional"/>
    <xs:attribute name="totalNumberofSuccessivePacketLoss" type="xs:unsignedLongVectorType"
        use="optional"/>
    <xs:attribute name="numberOfSuccessiveLossEvents" type="xs:unsignedLongVectorType"
        use="optional"/>
    <xs:attribute name="numberOfReceivedPackets" type="xs:unsignedLongVectorType" use="optional"/>
    <xs:attribute name="totalJitterDuration" type="xs:doubleVectorType" use="optional"/>
    <xs:attribute name="numberOfJitterEvents" type="xs:unsignedLongVectorType" use="optional"/>
    <xs:attribute name="framerate" type="xs:doubleVectorType" use="optional"/>
    <xs:attribute name="codecInfo" type="stringVectorType" use="optional"/>
    <xs:attribute name="codecProfileLevel" type="stringVectorType" use="optional"/>
    <xs:attribute name="codecImageSize" type="stringVectorType" use="optional"/>
    <xs:attribute name="averageCodecBitrate" type="doubleVectorType" use="optional"/>
    <xs:anyAttribute processContents="skip"/>
</xs:complexType>

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

<xs:simpleType name="unsignedLongVectorType"
    <xs:list itemType="xs:unsignedLong"/>
</xs:simpleType>

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

</xs:schema>
```

5.3.2.3.3.2 Example XML for the QoE Report

The example shows a QoE report for a streaming session.

```
<?xml version="1.0" encoding="UTF-8"?>
<receptionReport xmlns="urn:3gpp:metadata:2009:PSS:receptionreport"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="urn:3gpp:metadata:2009:PSS:receptionreport receptionreport.xsd">
    <statisticalReport
        clientId="clientID"
        serviceURI="xxx.example.com"
        serviceId="serviceID">
        <qoeMetrics
            numberOfRebufferingEvents="0 1 0"
            initialBufferingDuration="3.213"
            totalRebufferingDuration="0 1.23 0"
            contentAccessTime="2.621"
            sessionStartTime="1219322514"
            sessionStopTime="1219322541">
            bufferDepth="3.571 2.123 2.241"
            allContentBuffered="false">
            <medialevel_qoeMetrics
                sessionId="10.50.65.30:5050"
                framerate="15.1 14.8 15.0"
                t="false"
                numberOfSuccessiveLossEvents="5 0 3"
                numberOfCorruptionEvents="6 5 2"
```

```
                numberOfJitterEvents="0 1 0"
                totalCorruptionDuration="152 234 147"
                totalNumberofSuccessivePacketLoss="25 0 6"
                numberOfReceivedPackets="456 500 478"
                codecInfo="H263-2000/90000 = ="
                codecProfileLevel="profile=0;level=45 = ="
                codecImageSize="176x144 = ="
                averageCodecBitRate="124.5 128.0 115.1"
                totalJitterDuration="0 0.346 0"/>
        </qoeMetrics>
    </statisticalReport>
</receptionReport>
```

5.3.2.4 Video buffering headers

The following header fields are specified for the response of an RTSP PLAY request only:

- x-predecbufsize:<size of the pre-decoder buffer>
- x-initpredecbufperiod:<initial pre-decoder buffering period>
- x-initpostdecbufperiod:<initial post-decoder buffering period>
- 3gpp-videopostdecbufsize:<size of the video post-decoder buffer>

The header fields "x-predecbufsize", "x-initpredecbufperiod", "x-initpostdecbufperiod", and "3gpp-postdecbufsize" have the same definitions as the corresponding SDP attributes (see clause 5.3.3.2) "X-predecbufsize", "X-initpredecbufperiod", "X-initpostdecbufperiod", and "3gpp-postdecbufsize", respectively, with the exception that the RTSP video buffering header fields are valid only for the range specified in the RTSP PLAY response.

For H.263 and MPEG-4 Visual, the usage of these header fields is specified in Annex G.

For H.264 (AVC), PSS servers shall include these header fields in an RTSP PLAY response whenever the values are available in the 3GP file used for the streaming session. If the values are not available in the 3GP file, it is optional for the servers to signal the parameter values in RTSP PLAY responses.

5.3.3 SDP

5.3.3.1 General

RTSP requires a presentation description. SDP shall be used as the format of the presentation description for both PSS clients and servers. PSS servers shall provide and PSS clients interpret the SDP syntax according to the SDP specification [6] and appendix C of [5]. The SDP delivered to the PSS client shall declare the media types to be used in the session using a codec specific MIME media type for each media. MIME media types to be used in the SDP file are described in clause 5.4 of the present document.

The SDP [6] specification requires certain fields to always be included in an SDP file. Apart from this a PSS server shall always include the following fields in the SDP:

- "a=control:" according to clauses C.1.1, C.2 and C.3 in [5];
- "a=range:" according to clause C.1.5 in [5];
- "a=rtpmap:" according to clause 6 in [6];
- "a=fmtp:" according to clause 6 in [6].

When an SDP document is generated for media stored in a 3GP file, each control URL defined at the media-level 'a=control:' field shall include a stream identifier in the last segment of the path component of the URL. The value of the stream id shall be defined by the track-ID field in the track header (tkhd) box associated with the media track. When a PSS server receives a set-up request for a stream, it shall use the stream identifier specified in the URL to map the request to a media track with a matching track-ID field in the 3GP file. Stream identifiers shall be expressed using the following syntax:

streamIdentifier = <stream-id-token>"="<stream-id>

stream-id-token = 1*alpha

stream-id = 1*digit

The bandwidth field in SDP is needed by the client in order to properly set up QoS parameters. Therefore, a PSS server shall include the 'b=AS:' and "b=TIAS:" and "a=maxprate" [93] fields at the media level for each media stream in SDP, and should include "b=TIAS" and "a=maxprate" at session level. A PSS client shall interpret all of these fields. If both bandwidth modifiers are present, "b=TIAS" should be used; however it may be missing in content produced according to earlier releases. When a PSS client receives SDP, it should ignore the session level 'b=AS:' parameter (if present), and instead calculate session bandwidth from the media level bandwidth values of the relevant streams. If "b=TIAS" and "a=maxprate" is present at session level, it should be used in preference over the media level values, as session level can provide a more accurate description of the needed session bandwidth when aggregating several media streams together. A PSS client shall also handle the case where the bandwidth parameters are not present, since this may occur when connecting to a Release-4 server.

Note that for RTP based applications, "b=AS:" gives the RTP "session bandwidth" (including UDP/IP overhead) as defined in section 6.2 of [9].

The bandwidth for RTCP traffic shall be described using the "RS" and "RR" SDP bandwidth modifiers, as specified by [55]. The "RS" SDP bandwidth modifier indicates the RTCP bandwidth allocated to the sender (i.e. PSS server) and "RR" indicates the RTCP bandwidth allocated to the receiver (i.e. PSS client). A PSS server shall include the "b=RS:" and "b=RR:" fields at the media level for each media stream in SDP, and a PSS client shall interpret them. A PSS client shall also handle the case where the bandwidth modifier is not present according to section 3 of [55], since this may occur when connecting to a Release-4 server.

There shall be a limit on the allowed RTCP bandwidth for senders and receivers in a session. This limit is defined as follows:

- 4000 bps for the RS field (at media level);
- 5000 bps for the RR field (at media level).

In Annex A.2.1 an example SDP in which the limit for the total RTCP bandwidth is 5% of the session bandwidth is presented.

Media which has an SDP description that includes an open ended range (format=startvalue-) in any time format in the SDP attribute "a=range", e.g. "a=range: npt=now-", or "a=range: clock=20030825T152300Z-", shall be considered media of unknown length. Such a media shall be considered as non-seekable, unless other attributes override this property.

The "t=", "r=", and "z=" SDP parameters are used to indicate when the described session is active and can be used to filter out obsolete SDP files. PSS clients and servers shall support "t=", "r=", and "z=" as specified in [6]. The "a=etag" parameter may additionally be used to identify SDP validity. PSS clients should support "a=etag" as specified in [5].

When creating an SDP for a streaming session, one should try to come up with the most accurate estimate of time that the session is active. The "t=", "r=", and "z=" SDP parameters are used for this purpose, i.e., to indicate when the described session is active. If the time at which a session is active is known to be only for a limited period, the "t=", "r=", and "z=" attributes should be filled out appropriately (per [6], the "t=" shall be sent and usually contains non-zero values, possibly using the "r=" and "z=" parameters). If the stop-time is set to zero, the session is not bounded, though it will not become active until after the start-time. If the start-time is also zero, the session is regarded as permanent. A session should only be marked as permanent ("t=0 0") if the session is going to be available for a significantly long period of time or if the start and stop times are not known at the time of SDP file creation. Recommendations for what is considered a significant time is present in the SDP specification [6].

IPv6 addresses in SDP descriptions shall be supported according to RFC 4566 [6].

NOTE: The SDP parsers and/or interpreters shall be able to accept NULL values in the 'c=' field (e.g. 0.0.0.0 in IPv4 case). This may happen when the media content does not have a fixed destination address. For more details, see Section C.1.7 of [5] and Section 6 of [6].

5.3.3.2 Additional SDP fields

The following additional media level SDP fields are defined for PSS:

- "a=X-predecbufsize:<size of the hypothetical pre-decoder buffer>"
  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation (see clause 10.2) is not in use, this gives the suggested size of the Annex G hypothetical pre-decoder buffer in bytes.

  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is in use, this gives the suggested minimum size of a buffer (hereinafter called the pre-decoder buffer) that is used to smooth out transmit time variation (compared to flat-bitrate transmission scheduling) and video bitrate variation.

  If the field is an attribute for an H.264 (AVC) stream, the H.264 (AVC) bitstream is constrained by the value of "CpbSize" equal to X-predecbufsize * 8 for NAL HRD parameters, as specified in [90]. For the VCL HRD parameters, the value of "CpbSize" is equal to X-predecbufsize * 40 / 6. The value of "X-predecbufsize" for H.264 (AVC) streams shall be smaller than or equal to 1200 * MaxCPB, in which the value of "MaxCPB" is derived according to the H.264 (AVC) profile and level of the stream, as specified in [90]. If "X-predecbufsize" is not present for an H.264 (AVC) stream, the value of "CpbSize" is calculated as specified in [90].

- "a=X-initpredecbufperiod:<initial pre-decoder buffering period>"
  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is not in use, this gives the required initial pre-decoder buffering period specified according to Annex G. Values are interpreted as clock ticks of a 90-kHz clock. That is, the value is incremented by one for each 1/90 000 seconds. For example, value 180 000 corresponds to a two second initial pre-decoder buffering.

  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is in use, this gives the suggested minimum greatest difference in RTP timestamps in the pre-decoder buffer after any de-interleaving has been applied. Note that X-initpredecbufperiod is expressed as clock ticks of a 90-kHz clock. Hence, conversion may be required if the RTP timestamp clock frequency is not 90 kHz.

  If the field is an attribute for an H.264 (AVC) stream, the H.264 (AVC) bitstream is constrained by the value of the nominal removal time of the first access unit from the coded picture buffer (CPB), $t_{r,n}(0)$, equal to "X-initpredecbufperiod" as specified in [90]. If "X-initpredecbufperiod" is not present for an H.264 (AVC) stream, $t_{r,n}(0)$ shall be equal to the earliest time when the first access unit in decoding order has been completely received.

- "a=X-initpostdecbufperiod:<initial post-decoder buffering period>"
  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is not in use, this gives the required initial post-decoder buffering period specified according to Annex G. Values are interpreted as clock ticks of a 90-kHz clock.

  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is in use, this gives the initial post-decoder buffering period assuming that the hypothetical decoding and post-decoder buffering model given in points 5 to 10 in clause G.3 of Annex G would be followed. Note that the operation of the post-decoder buffer is logically independent from rate adaptation and is used to compensate non-instantaneous decoding of pictures.

  If the field is an attribute for an H.264 (AVC) stream, the H.264 (AVC) bitstream is constrained by the value of dpb_output_delay for the first decoded picture in output order equal to "X-initpostdecbufperiod" as specified in [90] assuming that the clock tick variable, t_c, is equal to 1 / 90 000. If "X-initpostdecbufperiod" is not present for an H.264 (AVC) stream, the value of dpb_output_delay for the first decoded picture in output order is inferred to be equal to 0.

- "a=X-decbyterate:<peak decoding byte rate>"
  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is not in use, this gives the peak decoding byte rate that was used to verify the compatibility of the stream with Annex G. Values are given in bytes per second.

  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is in use, "X-decbyterate" has no meaning.

  This field shall not be present for H.264 (AVC) streams.

- "a=3gpp-videopostdecbufsize:<size of the video post-decoder buffer>"
  This attribute may be present for H.264 (AVC) streams and it shall not be present for other types of streams. If the attribute is present, the H.264 (AVC) bitstream is constrained by the value of "max_dec_frame_buffering" equal to Min(16, Floor(3gpp-videopostdecbufsize / (PicWidthInMbs * FrameHeightInMbs * 256 *

ChromaFormatFactor))) as specified in [90]. If "3gpp-videopostdecbufsize" is not present for an H.264 (AVC) stream, the value of "max_dec_frame_buffering" is inferred as specified in [90].

If none of the attributes "a=X-predecbufsize:", "a=X-initpredecbufperiod:", "a=X-initpostdecbufperiod:", and "a=x-decbyterate:" is present for an H.263 or MPEG-4 Visual stream, clients should not expect a packet stream according to Annex G. If at least one of the listed attributes is present for an H.263 or MPEG-4 Visual stream, and if the client does not choose the usage of bit-rate adaptation via RTSP as described in clause 5.3.2.2, the transmitted video packet stream shall conform to Annex G. If at least one of the listed attributes is present for an H.263 or MPEG-4 Visual stream, but some of the listed attributes are missing in an SDP description, clients should expect a default value for the missing attributes according to Annex G.

If the interleaved packetization mode of H.264 (AVC) is in use, attributes "a=X-predecbufsize:", "a=X-initpredecbufperiod:", "a=X-initpostdecbufperiod:", and "a=3gpp-videopostdecbufsize:" apply to an H.264 (AVC) bitstream when de-interleaving of the stream from transmission order to decoding order has been done.

The following media level SDP field is defined for PSS:

- "a=framesize:<payload type number> <width>-<height>"
  This gives the largest video frame size of H.263 streams.

The frame size field in SDP is needed by the client in order to properly allocate frame buffer memory. For MPEG-4 Visual streams, the frame size shall be extracted from the "config" information in the SDP. For H.264 (AVC) streams, the frame size shall be extracted from the sprop-parameters-sets information in the SDP. For H.263 streams, a PSS server shall include the "a=framesize" field at the media level for each stream in SDP, and a PSS client should interpret this field, if present. Clients should be ready to receive SDP descriptions without this attribute.

If this attribute is present, the frame size parameters shall exactly match the largest frame size defined in the video stream. The width and height values shall be expressed in pixels.

If integrity protection is supported, the following SDP attributes shall be supported by the client and server:

- "a=3GPP-Integrity-Key" according to annex K;
- "a=3GPP-SRTP-Config" according to Annex K;
- "a=3GPP-SDP-Auth" according to Annex K.

If RTP retransmission is supported, the following SDP attribute shall be supported by the client and server:

- "a=rtcp-fb" according to clause 4.2 in [57].

5.3.3.3 The 'alt' and 'alt-default-id' attributes

The client should interpret the following two media level attributes: "alt" and "alt-default-id". A client from earlier releases will ignore these attributes and can safely do so in a correctly formatted SDP. If the attributes are used by the server they shall be used in a way that makes them backward compatible. When interpreted, they define a number of alternatives from which the client can select the most appropriate one.

A non-extended SDP gives only one alternative for each media part (Annex A.1 Example 1). This is the default alternative for each media. The new SDP attributes defined here are used to modify the default attributes or to add new attributes to the default attributes thus creating new alternatives. Each alternative is numerically identified.

The alternative attribute "alt" is used to replace or add an SDP line to the default configuration. If the alternative attribute contains an SDP line, for which the type and the modifier already exist in the default alternative, the default must be replaced with the given line(s). In case there are multiple lines with the same type and modifier in the default alternative, all of the lines must be replaced. Multiple alternative lines can be used to modify the default alternative. The alternative lines that are used to form a certain alternative shall all carry the same numerical identifier (Annex A.1, Examples 2-4).

The alternative identifier is a unique identifier that points out a single alternative in one media declaration. The identifier must be unique between all media descriptions and their alternatives as it is used for creating combinations between different medias with the grouping attribute (see 5.3.3.4).

The default configuration is in itself a valid alternative. Therefore an attribute (alt-default-id) is defined that assigns an alternative identifier to the default alternative. This identifier can then be used with the grouping attribute (see 5.3.3.4) to create combinations of alternatives from different medias.

The alternative attribute is defined below in ABNF from RFC 4234 [53]. The SDP line is any SDP line allowed at media level except "m=".

alt = "a" "=" "alt" ":" alt-id ":" SDP-line CRLF

SDP-line = <type>=<value> ; See RFC 4566 [6]

alt-id = 1*DIGIT ; unique identifier for the alternative in whole SDP.

To be able to assign an alternative ID to the default alternative, the following identification attribute is defined.

alt-default-id = "a" "=" "alt-default-id" ":" alt-id CRLF

5.3.3.4 The session level grouping attribute, 'alt-group'

The client should handle the following attribute: "alt-group". A client from earlier releases will ignore this attribute and can safely do so. When interpreted, it defines a number of grouping alternatives from which the client can select the most appropriate one. The identifiers defined in 5.3.3.3 are used together with the "alt-group" attribute to create combinations consisting of, e.g., one audio and one video alternative.

A grouping attribute is used to recommend certain combinations of media alternatives to the client. There may be more than one grouping attribute at the session level as long as they are for different grouping types and subtypes.

alt-group = "a" "=" "alt-group" ":" alt-group-type ":" alt-group-subtype ":" alt-grouping *(";" alt-grouping) CRLF

alt-group-type = token ; "token" defined in RFC 4566 [6]

alt-group-subtype = token

alt-grouping = grouping-value "=" alt-id *("," alt-id)

grouping-value = token

The alt-group attribute gives one or more combinations of alternatives through their IDs. Each grouping shall be given a grouping value. The grouping value is used to determine if the alternatives within the grouping suit the client. New types and subtypes can be added later.

The following grouping types and subtypes are defined:

- Type: BW, Subtype: All modifiers defined for the SDP "b=" attribute at session and media level. See www.IANA.org for current list of registered attributes.

  Grouping value: The bandwidth value defined for that modifier calculated over all the alternatives grouped together in that grouping. For SDP bandwidth modifiers defined at session level the value shall be calculated according to its rule over the alternative part of the grouping. For media-level-only modifiers, the grouping value shall be calculated as a sum of the media-level values in the grouped alternatives. For TIAS [93] the bandwidth value alone is not sufficient to provide a receiver with sufficient information to make a decision. The SDP attribute "maxprate" is also needed. To provide this information in the grouping-value the following syntax shall be used: <bit-rate>_<maxprate>, where <bit-rate> is the bit-rate value for TIAS and <maxprate> is the maxprate value corresponding to the SDP attribute.

  Grouping recommendations: Each grouping should only contain one alternative from each media type. There is no need to give groupings for all combinations between the media alternatives, rather it is strongly recommended to only give the most suitable combinations (Annex A.1 Example 5). The client can use the bandwidth values of the grouping to estimate the minimum, guaranteed or maximum bandwidth that will be needed for that session.

- Type: LANG Subtype: RFC3066

  Grouping value: A language tag as defined by RFC 3066 [54]. The grouping MUST contain all media alternatives, which support that language tag.

Grouping recommendations: It is recommended that other mechanisms, like user profiles if existing, are primarily used to ensure that the content has language suitable for the user (Annex A.1, Example 6).

Se also Annex A1, Examples 7 through 16. In the examples all three new attributes "alt", "alt-default-id" and "alt-group" are used.

5.3.3.5 The bit-rate adaptation support attribute, '3GPP-Adaptation-Support'

To signal the support of bit-rate adaptation, a media level only SDP attribute is defined in ABNF [53]:

sdp-Adaptation-line = "a" "=" "3GPP-Adaptation-Support" ":" report-frequency CRLF

report-frequency = NonZeroDIGIT [DIGIT]

NonZeroDIGIT = %x31-39 ;1-9

A server implementing rate adaptation shall signal the "3GPP-Adaptation-Support" attribute in its SDP.

A client receiving an SDP description where the SDP attribute "3GPP-Adaptation-Support" is present knows that the server provides rate adaptation. The client, if it supports bit-rate adaptation, shall then in its subsequent RTSP signalling use the '3GPP-Adaptation' header as defined in clause 5.3.2.2, as well as the RTCP Next Application Data Unit (NADU) APP packet for reporting the next unit to be decoded, as defined in clause 6.2.3.2.

The SDP attribute shall only be present at the media level. The report frequency value, which shall be larger than zero, indicates to the client that it shall include a NADU APP packet in a compound RTCP packet no less often than the interval specified by report-frequency, except prior to receipt of RTP media packets, when the client is unable to generate a valid NADU APP packet. For example, if this value is 3, the client shall send the NADU APP packet in at least every 3rd RTCP packet.

5.3.3.6 The Quality of Experience support attribute, "3GPP-QoE-Metrics"

PSS servers using QoE-Metrics in a session shall use SDP to initiate the QoE negotiation. The reason why SDP is needed is to support the use cases where SDP is distributed through other methods than RTSP DESCRIBE, e.g. WAP, HTTP or email. A new SDP attribute, which can be used either at session or media level, is defined below in ABNF [53] based on RFC 4566 [6]:

QoE-Metrics-line = "a" "=" "3GPP-QoE-Metrics:" att-measure-spec *("," att-measure-spec)) CRLF

att-measure-spec = Metrics ";" Sending-rate [";" Measure-Range] *([";" Parameter-Ext])

Metrics = as defined in clause 5.3.2.3.1.

Sending-Rate = as defined in clause 5.3.2.3.1.

Measure-Range = as defined in clause 5.3.2.3.1.

Parameter-Ext = as defined in clause 5.3.2.3.1.

PSS servers using QoE-Metrics in a session shall use this attribute to indicate that QoE metrics are supported and will be sent. When present at session level, it shall only contain metrics that apply to the complete session. When present at media level, it shall only contain metrics that are applicable to individual media. The URI that is used in the specification of the RTSP header "3GPP-QoE-Metrics:" is implicit by the RTSP control URI (a=control).

5.3.3.7 The asset information attribute, "3GPP-Asset-Information"

This asset information attribute is defined to transmit asset information in SDP. The attribute is defined ABNF [53]:

3GPP-Assets-Info = "a" "=" "3GPP-Asset-Information:" Asset 0*("," Asset) CRLF

Asset = ("{" "url" "=" <">URL<"> "}") / ("{"AssetName "=" AssetBox "}")

URL = as defined in [60]

AssetName = "Title" / "Description" / "Copyright" / "Performer" / "Author" / "Genre" / "Rating" /

"Classification" / "Keywords" / "Location" / "Album" / "RecordingYear" / asset-extension

asset-extension = 1*((0x01..0x09) / 0x0b / 0x0c / (0x0e..0x1f) / (0x21..0x2b) / (0x2d..0x3c) / (0x3e..0x7a) /

0x7c / (0x7d..0xff)) ;any byte except SP, NUL, CR, LF , "=", ",", "{" or "}"

AssetBox = Base64 encoded version [69] of any asset box as defined in Clause 8 of [50].

This SDP attribute can be present at session level, media level or both. Multiple instances of the attribute are allowed.

The resource referenced by the URL can be any pre-formatted data, e.g. an XHTML page or XML file, containing any asset information. It is up to the client"s capability and user"s preference to render the information pointed by the URL.

Example 17 in Clause A.1 shows an SDP file that includes the "3GPP-Asset-Information" attribute.

5.3.3.8 OMA-DM Configuration of QoE Metrics

As an optional alternative to configure the QoE reporting for each session (i.e. via SDP/RTSP), OMA-DM can be used to specify the default QoE configuration. If such a default QoE configuration has been specified, it shall be used by the terminal for all subsequent PSS sessions where no session-specific QoE configuration is received. QoE reporting based on the default OMA configuration shall always be done over HTTP with the XML reporting format. If the PSS client does not support HTTP reporting it shall not send default QoE reports.

Any session-specific QoE configuration received shall always have higher priority, and will in such cases override any default OMA-DM QoE configuration for that session.

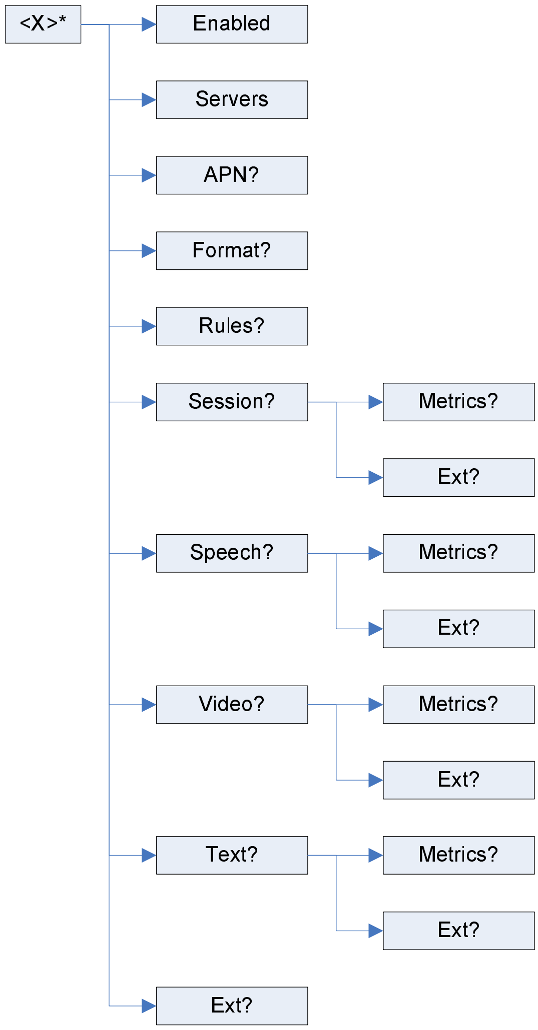
For OMA-DM QoE configuration the parameters are specified according to the following Managed Object (MO). Version numbering is included for possible extension of the MO.

The Management Object Identifier shall be: urn:oma:mo:ext-3gpp-pssqoe:1.0.

Protocol compatibility: The MO is compatible with OMA Device Management protocol specifications, version 1.2 and upwards, and is defined using the OMA DM Device Description Framework as described in the Enabler Release Definition OMA-ERELD _DM-V1_2 [100].

5.3.3.8.1 QoE metrics reporting management object

The following nodes and leaf objects shall be contained under the 3GPP_PSSQOE node if a PSS client supports the feature described in this clause (information of DDF for this MO is given in Annex P):



Node: /<X>

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

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

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

/<X>/Servers

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

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

- Values: URI of the servers to receive the QoE report.

/<X>/Enabled

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

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

/<X>/APN

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

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

/<X>/Format

This leaf specifies the format of the report and if compression (Gzip XML) [xx] is used.

- Occurrence: ZeroOrOne
- Format: chr
- Minimum Access Types: Get
- Values: 'XML', 'GZIPXML'.

/<X>/Rules

This leaf provides in textual format the rules used to decide whether metrics are to be reported to the QoE metrics report server. The syntax and semantics of this leaf are defined in sub-clause 5.3.3.8.2.

- Occurrence: ZeroOrOne
- Format: chr
- Minimum Access Types: Get
- Values: See clause 5.3.3.8.2.

/<X>/Ext

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

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

/<X>/Session

The Session node is the starting point of the session level QoE metrics definitions.

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

/<X>/Session/Metrics

This leaf provides in textual format the QoE metrics that need to be reported, the measurement frequency, the reporting interval and the reporting range. The syntax and semantics of this leaf are defined in clause 8.3.2.1.

- Occurrence: ZeroOrOne
- Format: chr
- Minimum Access Types: Get
- Values: see clause 8.3.2.1.

/<X>/Session/Ext

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

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

/<X>/Speech

The Speech node is the starting point of the speech/audio media level QoE metrics definitions.

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

/<X>/Speech/Metrics

This leaf provides in textual format the QoE metrics that need to be reported, the measurement frequency, the reporting interval and the reporting range. The syntax and semantics of this leaf are defined in clause 8.3.2.1.

- Occurrence: ZeroOrOne
- Format: chr
- Minimum Access Types: Get
- Values: see clause 8.3.2.1.

/<X>/Speech/Ext

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

- Occurrence: ZeroOrOne

- Format: node
- Minimum Access Types: Get

/<X>/Video

The Video node is the starting point of the video media level QoE metrics definitions.

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

/<X>/Video/Metrics

This leaf provides in textual format the QoE metrics that need to be reported, the measurement frequency, the reporting interval and the reporting range. The syntax and semantics of this leaf are defined in clause 8.3.2.1.

- Occurrence: ZeroOrOne
- Format: chr
- Access Types: Get
- Values: see clause 8.3.2.1.

/<X>/Video/Ext

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

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

/<X>/Text

The Text node is the starting point of the timed-text media level QoE metrics definitions.

- Occurrence: ZeroOrOne
- Format: node
- Minimum Access Types: Get
- Values: see clause 8.3.2.1.

/<X>/Text/Metrics

This leaf provides in textual format the QoE metrics that need to be reported, the measurement frequency, the reporting interval and the reporting range. The syntax and semantics of this leaf are defined in clause 8.3.2.1.

- Occurrence: ZeroOrOne
- Format: chr
- Minimum Access Types: Get
- Values: see clause 8.3.2.1.

/<X>/Text/Ext

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

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

5.3.3.8.2 QoE reporting rule definition

This clause defines the syntax and semantics of a set of rules which are used to reduce the amount of reporting to the QoE metrics report server. The syntax of the metrics reporting rules is defined below:

- QoE-Rule = "3GPP-QoE-Rule" ":" rule-spec *("," rule-spec)
- rule-spec = rule-name [";" parameters]
- rule-name = "LimitSessionInterval" / "SamplePercentage"
- parameters = parameter *(";" parameter)
- parameter = Param-Name ["=" Param-Value]
- Param-Name = 1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7e) ;VCHAR except ";", ",", "{" or "}"
- Param-Value = (1*DIGIT ["." 1*DIGIT]) / (1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7c / 0x7e))

The semantics of the rules and the syntax of its parameters are defined below:

The *SamplePercentage* rule can be used to set a percentage sample of calls which should report reception. This can be useful for statistical data analysis of large populations while increasing scalability due to reduced total uplink signalling. The *sample_percentage* parameter takes on a value between 0 and 100, including the use of decimals. It is recommended that no more than 3 digits follow a decimal point (e.g. 67.323 is sufficient precision).

When the *SamplePercentage* rule is not present or its *sample_percentage* parameter value is 100 each PSS client shall send metric report(s). If the *sample_percentage* value is less than 100, the UE generates a random number which is uniformly distributed in the range of 0 to 100. The UE sends the reception report when the generated random number is of a lower value than the *sample_percentage* value.

The *LimitSessionInterval* rule is used to limit the time interval between consecutive sessions that report metrics. The *min_interval* parameter for this rule indicates the minimum time distance, in seconds, between the start of two sessions that are allowed to report metrics. When this rule is absent there is no limitation on the minimum time interval.

In case multiple rules are defined in the Management Object, the PSS client should only report metrics when all individual rules evaluate to true (i.e. the rules are logically ANDed). In case no rules are present the PSS client should always report metrics (see also clause xx for metrics reporting procedures).

An example for a QoE metric reporting rule is shown below:

3GPP-QoE-Rule:SamplePercentage;sample_percentage=10.0,LimitSessionInterval;min_interval=300

This example rule defines that only 10% of the sessions shall report, with the minimum time interval between the start times of two consecutive sessions that report metrics to be 5 minutes.

5.4 MIME media types

For continuous media the following MIME media types shall be used:

- AMR narrow-band speech codec (see sub-clause 7.2) MIME media type as defined in [11];
- AMR wideband speech codec (see sub-clause 7.2) MIME media type as defined in [11];
- Extended AMR-WB codec (see sub-clause 7.3) MIME media type as defined in [85];

- Enhanced aacPlus and MPEG-4 AAC audio codecs (see clause 7.3) MIME media type as defined in RFC 3016 [13].
  The following applies to servers when this MIME type is used in SDP:

  1. Configuration information is exclusively carried out-of-band in the SDP 'config' parameter; this shall be signaled by sending 'cpresent=0'.

  2. A PSS server serving implicitly signaled Enhanced aacPlus content shall include 'SBR-enabled=1' in the 'a=fmtp' line; it shall include 'SBR-enabled=0' if it serves plain AAC content.

  3. A PSS server serving explicitly signaled content is recommended not to include the 'SBR-enabled' parameter in the 'a=fmtp' line.

  Therefore, the following applies to terminals:

  1. The rtpmap rate parameter should not be considered definitive of the sampling rate (though it is, of course, definitive of the timescale of the RTP timestamps).

  2. If explicit signaling is in use, the StreamMuxConfig contains both the core AAC sampling rate and the SBR sampling rate. The appropriate output sampling rate may be chosen dependant on Enhanced aacPlus support.

  3. If explicit signalling is not in use and no SBR-enabled parameter is present, the StreamMuxConfig contains the AAC sampling rate and the appropriate output sampling rate may be set to this indicated rate.

  4. If explicit signalling is not in use and the SBR-enabled parameter is present, terminals supporting Enhanced aacPlus should set the output sampling rate to either the core AAC sampling rate as indicated in the StreamMuxConfig [21] (where 'SBR-enabled' is set to '0') or twice the indicated rate (where 'SBR-enabled' is set to '1');

- MPEG-4 Visual video codec (see sub-clause 7.4) MIME media type as defined in RFC 3016 [13]. When used in SDP the configuration information shall be carried outband in the "config" SDP parameter and inband as stated in RFC 3016. As described in RFC 3016, the configuration information sent inband and the config information in the SDP shall be the same except that first_half_vbv_occupancy and latter_half_vbv_occupancy which, if exist, may vary in the configuration information sent inband;

- H.263 [22] video codec (see sub-clause 7.4) MIME media type as defined in clause 8.1.2 of [14]. In order to guarantee backward compatibility with earlier Releases (before Release 7), MIME parameters other than 'profile' and 'level' should not be used;

- H.264 (AVC) [90] video codec (see sub-clause 7.4) MIME media type as defined in [92];

- 3GPP timed text format [51] MIME media type as defined in sub-clause 7.1 of [80];

- OMA DRM protected streaming media MIME media type as defined in clause K.1.4 in Annex K;

- RTP retransmission payload format MIME media types as defined in clause 8 of [81];

- DIMS MIME media type as defined in [98].

MIME media types for JPEG, GIF, PNG, SP-MIDI, Mobile DLS, Mobile XMF, SVG, timed text, 3GP and XHTML can be used in the "Content-type" field in HTTP, "content_type" field in the item information box of 3GP files, and in the "type" attribute in SMIL 2.0, SVG Tiny 1.2 and DIMS. The following MIME media types shall be used for these media:

- JPEG (see sub-clause 7.5) MIME media type as defined in [15];

- GIF (see sub-clause 7.6) MIME media type as defined in [15];

- PNG (see sub-clause 7.6) MIME media type as defined in [38];

- SP-MIDI (see sub-clause 7.3A) MIME media type as defined in clause C.2 in Annex C of the present document;

- DLS MIME media type to represent Mobile DLS (see sub-clause 7.3A) as defined in [97];

- Mobile XMF (see sub-clause 7.3A) MIME media type as defined in clause C.3 in Annex C of the present document;
- SVG (see sub-clause 7.7) MIME media type as defined in [42];
- XHTML (see sub-clause 7.8) MIME media type as defined in [16];
- Timed text (see sub-clause 7.9) MIME media type as defined in [79];
- 3GP files (see sub-clause 7.10) MIME media type as defined in [79].

MIME media type used for SMIL files shall be according to [31] and for SDP files according to [6].

NOTE: The 3GP MIME media type [79] is used for all 3GP files, including 3GP files carrying timed text, DIMS, images, etc.

5.5 Extension for Fast Content Switching and Start-up

5.5.1 Introduction

Applications which are built on top of packet switched streaming (PSS) services are classified into on-demand and live information delivery applications. This clause defines procedures to allow faster start up and switching of content for both on-demand and live applications by reducing the client/server interactions to a minimum. Additionally, clients are enabled to reuse the existing RTSP control session and RTP resources while switching to new content.

5.5.2 Extensions to RTSP 1.0

5.5.2.1 Introduction

Various general RTSP extensions are required for support of fast content start-up and switching. These extensions must be implemented by PSS clients and servers wishing to support any of these features.

The following new RTSP feature tags are defined:

- '3gpp-pipelined' feature-tag, section 5.5.3
- '3gpp-switch' feature-tag, section 5.5.4.3
- '3gpp-switch-req-sdp' feature-tag, section 5.5.4.4
- '3gpp-switch-stream' feature-tag, section 5.5.4.5

In addition the following new RTSP header fields are defined:

- 'Switch-Stream' header, section 5.5.4.2
- 'SDP-Requested' header, section 5.5.4.4
- 'Pipelined-Requests' header, section 5.5.3

5.5.2.2 Capability Handling

5.5.2.2.1 Introduction

The PSS client shall determine the PSS server''s capabilities and indicate its own capabilities to the server using the 'Supported' header field (see clause 5.5.2.2.2) as early as possible (e.g. with the DESCRIBE request).

The 'Require' header field is used as defined in RFC 2326 [5] to ensure that a certain feature is supported by the PSS server. An unsupported feature shall cause the containing request to fail with a 551 reply code and 'Option not supported' as the reason. The 551 reply shall also include the unsupported features in the 'Unsupported' header field.

The 'Require' header field should not be used for probing support for features but rather to make sure that a specific request is executed correctly with the specified features.

5.5.2.2.2 Definition of the 'Supported' RTSP Header Field

PSS clients and servers must support the 'Supported' RTSP header field.

The 'Supported' header field enumerates all the extensions supported by the client or server using feature tags. The header carries the extensions supported by the message sending entity. The 'Supported' header may be included in any request. When present in a request, the server shall respond with its corresponding 'Supported' header.

If an RTSP request includes a 'Supported' header but the corresponding response does not include this header field, then the PSS client shall assume that the PSS server does not support any of the indicated features

Note, the 'Supported' header must be included in error as well as success responses.

The Supported header field contains a list of feature-tags, described in Section 5.5.2.1, that are understood by the client or server.

Example:

```
C->S:   OPTIONS rtsp://3gpp.org/ RTSP/1.0
        CSeq: 1
        Supported: 3gpp-pipelined

S->C:   RTSP/1.0 200 OK
        CSeq: 1
        Public: OPTIONS, DESCRIBE, SETUP, PLAY, PAUSE, TEARDOWN
        Supported: 3gpp-pipelined, 3gpp-switch, 3gpp-switch-req-sdp, 3gpp-switch-stream
```

5.5.2.3 SSRC in the 'RTP-Info' RTSP Header Field

The "RTP-Info" response header field is used to set RTP-specific parameters in the PLAY response. For streams using RTP as transport protocol the "RTP-Info" header is always part of a 200 response to PLAY (as defined in Annex A.2.1). In addition to the parameters defined in RFC 2326 [5], the "RTP-Info" header may also include a synchronization source (SSRC) parameter.

Note. The SSRC parameter is mandatory for some content switching procedures defined in clause 5.5.4.

The SSRC parameter gives the synchronization source (SSRC) of the RTP flow to which the RTP timestamp and sequence number apply. It is only possible to describe one synchronization source (SSRC) per media resource.

After a fast content switch (FCS) the SSRC source used on a specific RTP session may change. In the event that the SSRC changes, it shall be included in the RTP-Info header. The PSS server shall change the SSRC value for a specific RTP session after a fast content switching operation is performed and

- if the payload type of the old and of the new media stream is the same but the media codec configuration is different, or
- if the mapping of the new media stream is otherwise unknown to the PSS client.

In case the SSRC remains unchanged after a content switch, the RTP sequence number and timestamp should be continuous and shall be monotonically increasing. Otherwise, a random RTP sequence number and timestamp should be used.

Further details on the usage of the SSRC are described in clause 5.5.3 or clause 5.5.4. Note the SSRC may only be included in the "RTP-Info" header, if the client has requested one of the features defined in clause 5.5.3 or clause 5.5.4.The "RTP-Info" header syntax in ABNF [53] is as follows:

```
RTP-Info         =  "RTP-Info" ":"rtsp-info-spec *("," rtsp-info-spec) CRLF

rtsp-info-spec   =  stream-url 1*parameter

stream-url       =  "url" "=" rtsp-url

parameter        =  ";" "ssrc" "=" 8HEX
```

/ ";" "seq" "=" 1*DIGIT

/ ";" "rtptime" "=" 1*DIGIT

5.5.2.4 Semantics of RTSP PLAY method

The queued PLAY functionality described in RFC 2326 [5] is removed. If a PLAY request is received for an RTSP session that is in the Playing State, then the server shall immediately execute the new PLAY request and terminate the old.

5.5.3 Start-up

In order to improve start-up times, a client may pipeline all necessary SETUP requests and the PLAY request. This allows streaming to begin with a single RTSP round trip if the client already has the SDP (or other adequate content description), or two round trips if it needs to first perform a DESCRIBE in order to receive the necessary information.

If the client intends to send upstream packets to ensure correctly open firewalls (also called port punching packets), then the client should not send a PLAY request until all SETUP responses are received. Pipelining of SETUP requests is still possible in this case.

If the client uses RTSP DESCRIBE to fetch the SDP from the server, then the client shall probe the server capabilities as described in clause 5.5.2.2 using the feature-tag value "3gpp-pipelined".

The client shall add the RTSP 'Require' header to all but the first pipelined RTSP SETUP request with the value '3gpp-pipelined". Note that the first RTSP SETUP request shall not use a 'Require' header. This will allow the PSS client to interoperate with minimal impact with older servers that do not support this feature.

Since the session does not yet exist when these pipelined messages are sent, a request header is defined which allows the client to inform the server that these messages are to be carried on the same session once it is created. Clients wishing to use pipelined start-up must implement the 'Pipelined-Requests' header in order to signal the session grouping to the server.

The syntax of the 'Pipelined-Requests' header is defined in ABNF [53] as follows:

Pipe-Hdr = "Pipelined-Requests" COLON startup-id

startup-id = 1*8DIGIT

The client should monitor whether the server behaves as declared.

A client unique 'startup-id' is required until the client receives the session ID. The 'startup-id' is unique for a particular TCP connection. Pipelined requests using this header must be sent on the same TCP connection. The method through which this ID is generated is to be decided by the client.

5.5.4 Fast Content Switching

5.5.4.1 Introduction

In most cases, a content switch can be initiated with a single RTSP request. In order to preserve interoperability with RTSP aware intermediate devices such as application layer gateways, PSS clients should ensure that SETUP requests and responses are sent for each RTP/RTCP port pair to be used. Once a port pair has been negotiated, it may be reused for subsequent content upon a switch.

5.5.4.2 'Switch-Stream' RTSP Header Field

The 'Switch-Stream' header field may be used in an RTSP PLAY request or an RTSP PLAY response message. It is used to describe the replacement of media streams after a content switch. The 'Switch-Stream' header field may be used with aggregated control and with media control URLs.

The 'Switch-Stream' header syntax in ABNF [53] is as follows:

Switch-Stream = "Switch-Stream" COLON switch-spec *(COMMA switch-spec) CRLF

```
switch-spec      = old-stream ";" new-stream
old-stream       = "old" "=" (DQ rtsp-url DQ) / (DQ DQ)
new-stream       = "new" "=" (DQ rtsp-url DQ) / (DQ DQ)
rtsp-url         = as defined in RFC 2326 [5]
DQ               =  %x22  ; US-ASCII double-quote mark (34)
LWS              =  [CRLF] 1*( SP / HT )
SWS              =  [LWS] ; sep whitespace
COMMA            =  * ( SP / HT ) "," SWS; comma
COLON            =  * ( SP / HT ) ":" SWS; colon
```

If both old media stream and new media stream URLs are indicated in the 'Switch-Stream' header field of a PLAY request from a PSS client to a PSS server, then the server shall interpret this as a request to replace the old media stream with the new media stream, hence reusing the transport parameters of the old media stream for the new media stream.

If the 'Switch-Stream' header field is included in a PLAY response from a PSS server to a PSS client, then this header informs the client about the media streams that are currently being streamed to the PSS client. The old media stream may be omitted in this case.

If only the new media stream URL is indicated in the 'Switch-Stream' header field of a PLAY request from a PSS client to a PSS server, then the PSS server shall interpret this as a request to switch to the new media stream. The PSS server decides the mapping. The PSS server shall indicate the SSRC of the new media stream in the RTP-Info of the reply, in order to enable the PSS client to locate the new stream.

If only the old stream URL is indicated in the 'Switch-Stream' header field of a PLAY request from a PSS client to a PSS server, then the PSS server shall interpret this as a request for complete removal of the specified media stream. The client and the server release the resources for this stream without explicit TEARDOWN signalling. The usage of the switch-stream header is defined in clauses 5.5.4.3, 5.5.4.4 and 5.5.4.7.

5.5.4.3 Switching to new content with available SDP

This clause defines all necessary PSS client and PSS server features for fast content switching where the UE already has the SDP for the new content locally available. The UE may have fetched the SDP file using RTSP DESCRIBE or HTTP GET or in any other method. Clients should assume that the herein defined fast content switching procedure is supported for all content items offered by this server. This PSS feature reduces the switching to new content to a single client-server interaction.

The feature-tag indicating this feature is '3gpp-switch'. The client should probe the server capabilities as early as possible in the communication using the '3gpp-switch' in the 'Supported' header as defined in 5.5.2.2.2. The client shall use the 'Require' header with this feature tag value, when requesting this behaviour from the server. The server shall use the PLAY method as defined in 5.5.2.4 with the '3gpp-switch' feature tag in the 'Require' header when the client requests this feature. Thus, the server replaces the current RTSP PLAY request by the new request resulting in a switch of streamed content.

When the PSS client wants to change the content of the RTSP session, the PSS client sends a PLAY request with the aggregated control URI of the new content to the PSS server. Note, the aggregated control URI is defined in the SDP file by the session level 'a=control:' attribute.

The PSS client shall add the media control URIs of the new streams in the 'Switch-Stream' header field to the RTSP PLAY method request. Whenever possible, the PSS client shall map the media control URIs of the same media type (e.g. audio or video) in the old content to the same media type of the new session. Note, this is only applicable for media types, which are present in the old and new content. The server includes always the 'Switch-Stream' header in the response. The 'Switch-Stream' header field is defined in 5.5.2.4.

If the SSRC have changed, then the server shall indicate the new SSRC values of the new media streams within the 'RTP-Info' header in the response. The SSRC entry for the 'RTP-Info' is defined in clause 5.5.2.3.

Note, if the new SDP contains more media components than the current session, the client may switch according to this section, describing the desired components in the 'Switch-Stream' header, and add the missing components using the method defined in 5.5.4.6.

If less media components are described in the new SDP than currently in use, the client and the server remove the component as defined in 5.5.4.7.

The entity-tag attribute ("a=etag" as specified in [5]) may be used in order to ensure that SDP information is used only if it is valid. When a PSS client requests a switch to content for which an entity-tag is available, it should include the 'If-Match' header and the etag value in the PLAY request. A PSS server shall validate the entity-tag in the If-Match headers prior to accepting the request. If the entity-tag is not valid, the server shall return either 412 (Precondition Failed) or if the client has previously communicated support for the "3gpp-switch-req-sdp" feature, the server may respond with a current SDP in an appropriate success response, as defined in Section 5.5.4.4.

5.5.4.4 Switching to new content without SDP

Clients should assume, that the here defined fast content switching procedure is supported for all content items offered by this server. The client uses the URL of the SDP file as content URL to describe the new content item.

Without an SDP or other adequate content description, the client is unable to specify the streams to which it wishes to subscribe. In order to initiate a content switch within a single RTSP round trip, the client may perform a PLAY request to initiate a switch via content URL without specifying individual streams. This allows the client to request that the server return the SDP, initiate a new session, setup all relevant media streams (or make an appropriate stream selection), and begin playback. The content URL used in the PLAY request is the same content URL used in a DESCRIBE. In order to signal that it wishes to receive the description and make a switch, the client shall include the 'SDP-Requested' header as defined below. This header is defined as follows:

SDP-Requested-Header = " SDP-Requested" COLON "1"

If a server receives a PLAY request and completes all actions successfully, the server responds with the SDP, Session-ID, RTP-Info, and a 'Switch-Stream' descriptor and begins streaming immediately. Whenever possible, the PSS server shall map the media control URIs of the same media type (e.g. audio or video) in the old content to the same media type of the new session. Note, this is only applicable for media types, which are present in the old and new content. The RTP-Info in the PLAY response must contain the SSRC for each stream as defined in 5.5.2.3. The server may issue a new session ID in the response, or it may re-use the existing session ID. The client must be prepared for either case.

If the server is not yet able to begin streaming, it responds with a 202 (Accepted) success code and with the SDP. The client may then perform a switch as described in 5.5.4.3 specifying the streams it would like to receive. This condition can occur if the server requires further client input regarding stream setup prior to beginning playback - for instance if the content requested contains multiple language switch groups and the server does not have the information necessary to choose a language.

If the server is not yet able to begin transmitting all the media streams, it can begin a subset of the streams and respond with a 206 (Partial Data) success code and the SDP. The 'Switch-Stream' header and the 'RTP-Info' header will indicate which streams have been selected for playback. The client may then add additional media components as described in 5.5.4.6.

If fewer media components are described in the new SDP than currently in use, then the server responds with a 200 (OK). The terminal shall remove the 'unused' media components as defined in clause 5.5.4.7.

The client and the server shall release the resources for the unused streams without explicit TEARDOWN signalling.

The feature tag '3gpp-switch-req-sdp' is defined to describe support for this feature. The client should probe the server capabilities as early as possible in the communication using the 'Supported' header as defined in 5.5.2.2 and shall use the 'Require' header with this feature tag value when requesting this behaviour from the server. The server shall use the PLAY method semantics defined in 5.5.2.4 when the client requests this feature.

5.5.4.5 Switching Media described in one SDP

Some content may be available for streaming in different representations. An example of such a use case is the live streaming of a sport event with multiple camera views. The SDP available at the receiver describes multiple options for one or several media types (e.g. video, audio, or subtitles). Upon initial setup of the session, the player (or the user) selects the preferred combination of the presentation to be consumed and sets up the corresponding media streams. At a later point, the user may trigger a switch to a different media stream carrying an alternative representation of the media.

The PLAY request is sent with the 'Switch-Stream' header field as defined in clause 5.5.4.2 indicating the URLs of both the old media stream and the new replacement stream. Upon receiving a PLAY request with a 'Switch-Stream' header field for an active session, a PSS server that supports this feature switches to the new media stream using the same transport parameters described in the initial SETUP request for the old media stream. After successfully processing the request, the PSS server shall reply with an 'RTP-Info' header indicating all active media streams in the changed session. The 'RTP-Info' header may include the SSRCs for each active media stream. The response may also include the 'Switch-Stream' header, indicating the stream switches that were successful. If the 'Switch-Stream' header field is not present in a successful response and the PSS server was identified to support the media switching functionality, the receiver should assume that all requested switches were successful.

The feature tag '3gpp-switch-stream' is defined to describe support for this feature. This feature tag is different than the feature tag '3gpp-switch' feature described in 5.5.4.3 indicating the support for content (aggregated stream) switch. The client should use the 'Require' header with this feature tag value when requesting this behaviour from the server. The server shall use the PLAY method semantics as defined in 5.5.2.4 when the client requests this feature. Note that several media streams of a presentation may be switched at the same time in a single PLAY request.

5.5.4.6 Adding Media Components to an ongoing session

It may happen that the new content stream consists of more media components than the ongoing content stream. In such a case, the client is recommended to switch to the new content with the already established resources and add further components afterwards.

The client should pipeline the setup requests for the new components after the content switching request (see clause 5.5.4). The client shall issue a PLAY request to start all addition media components without interrupting the existing. If the client and server support the "3gpp-pipelined" feature (see clause 5.5.3), then the client shall pipeline the PLAY request with the SETUP requests. The 'RTP-Info' header contains the synchronization information for all media components.

The session id value of the already established session shall be part of the SETUP request header to indicate the relation of the media component to the already established components.

5.5.4.7 Removing Media Components from an ongoing session

A PSS client wishing to terminate the streaming of a specific media stream shall send a PLAY request with a 'Switch-Stream' header indicating the URL of the media stream to be torn down as the old media stream. No URL for the new media stream should be specified.

Upon receiving a PLAY request with 'Switch-Stream' header field indicating that one or more media streams are to be terminated, the server shall stop streaming the indicated media streams and release the used UDP ports for this media component and free the associated resources. However, the other media streams should not be interrupted.

After successfully processing the request, the server shall reply with a success response message and a 'Session' header field, even if the session contains no more media streams.

The PSS client shall only use TEARDOWN to completely tear down the whole session.

5.6 Extension for Time-Shifting Support

5.6.1 Introduction

Time shifting functionality is designed to enhance the access to live streaming sessions. For this reason, the PSS server maintains a time-shift buffer for each live feed. The server side timeshift buffer allows the PSS client to pause live sessions and even navigate (rewind, fast forward) in the offered time-shift buffer range. A timeshift supporting PSS client, which is connected to a timeshift supporting PSS server is able to perform some or all of the following operations on timeshifted streaming sessions:

- Pause and resume the playout at a later point in time

- Start playout from (or seek to) a position in the stream that corresponds to a past time instant in the live streaming session
- Perform operations such as Fast and Slow Forward or Rewind (i.e. Trick Mode).

NOTE: the live streaming feed is not necessarily offered by the same PSS server as the time-shifted session. For example, the PSS server may be handling time-shifting services for a live feed accessible as an MBMS streaming service.

5.6.2 General Description

The PSS timeshift functionality requires the availability of one or more timeshift buffers on the server side. The size of the timeshift buffer is determined by the server. This specification does not limit the size of the timeshift buffer.

The timeshift buffer is defined by an upper and a lower range. New live data is added at the upper range to the timeshift buffer. If the timeshift buffer progresses as sliding window, then the server removes and discards data from the lower range of the timeshift buffer. The upper range of the timeshift buffer is also referred to as current recording time.

The PSS server announces the availability of the timeshift feature and describes the current state of the time-shift buffer in the RTSP SETUP response. The PSS server provides the current recording time and the current time-shift buffer ranges with each RTSP PLAY and PAUSE response messages. The server may report updated timeshift buffer ranges throughout the session via SET_PARAMETER messages and the client may request this information via GET_PARAMETER requests.

In the following, two PSS timeshift use-cases are described to clarify the operations of PSS timeshift.

Usecase A: A user starts a live session using 'range: npt=now-'. The user enters timeshift operation at a later point by pressing 'pause'. The PSS client stores the value of the 'current recording time' header, which is received with the pause response message to resume the session. The PSS client is aware of the server side timeshift buffer ranges.

Usecase B: A user may directly start a live session in a timeshift operation, for instance when changing from broadcast to reception of the same content. The PSS client may use absolute time representation (clock) to continue consuming the content at the same media position.

5.6.2a Extensions to RTSP 1.0

At least one PSS timeshift buffer is provided by the PSS server for at least one live stream. The PSS server may also provide individual timeshift buffers per RTSP session. The PSS server announces the availability of the timeshift feature and describes the current range of the time-shift buffer in the RTSP SETUP response.

The PSS client should always calculate current valid boundaries of the timeshift buffer.

The PSS timeshift feature may be used with either NPT or UTC time ranges. PSS clients that support time shifting shall support also UTC time in addition to NPT.

If the PSS server supports PSS timeshifting as defined in this section, then the new RTSP headers 3GPP-TS-CurrentRecording-Time and 3GPP-TS-Buffer shall be present in all server to client messages.

The Accept-Ranges header field may be used to check for the support of UTC time. The PSS server indicates the preferred media range format for time shifting in the 3GPP-TS-CurrentRecording-Time and 3GPP-TS-Buffer headers in the PLAY response. PLAY requests may contain the 'npt=now-' range indication to seek to the upper range of the time shift buffer. Other time shifting operations shall consistently use the same media range format that is indicated by the PSS server.

5.6.3 Accept-Ranges

The Accept-Ranges request and response-header field allows indication of the format supported in the Range header. The PSS client shall include the header in SETUP requests to indicate which formats it supports in PLAY and PAUSE responses and REDIRECT requests. The server shall include the header in the SETUP response and in any error response caused by an unaccepted range format, to indicate the formats supported for the resource indicated by the request URI.

This header has the following ABNF syntax:

Accept-Ranges = "Accept-Ranges" HCOLON acceptable-ranges CRLF

acceptable-ranges = range-unit *(COMMA range-unit)

range-unit = "npt" / "utc"

5.6.4 Signalling Time Shifting Ranges

In order to allow the PSS server to provide the PSS client with the current ranges of the time shift buffer, two new RTSP headers are defined.

The PSS client may start a PSS session already in 'timeshift mode'.
The PSS server provides the upper bound of the timeshift buffer with the '3GPP-TS-CurrentRecording-Time' header. The PSS client should not request playback beginning beyond the current recording time (i.e. no future playback times). If a PSS server receives a PLAY request outside of the time shift buffer range, the PSS server should handle the request as a request for the appropriate buffer boundary time. In case of a range request exceeding the range end time, this will be the end time; in case of a range request beginning earlier than the buffer start time, this will be the start time. The actual range streamed will then be reported in the 200 OK response message.

The PSS server describes the current available range for PSS time shifting with the '3GPP-TS-Buffer' header. This header may use one of three descriptive formats, depending on the current state of the time-shift buffer. The client is responsible for keeping track of the available timeshift buffer boundaries.

The 'buffer-depth' parameter indicates the depth of the timeshift buffer in seconds. When this format is indicated, the timeshift buffer is constantly filled to the specified depth; the timeshift buffer is progressed as a sliding window. The PSS client calculates together with the information from the '3GPP-TS-CurrentRecording-Time' header the lower and upper range of the timeshift buffer. The PSS client shall continuously update the upper and lower boundary of the timeshift buffer.

The 'interval' parameter can either indicate a closed range (two value) or an open range (one value) indicating absolute times for the timeshift buffer ranges. No timeshift data is available earlier than the left range value of the 3GPP-TS-Buffer parameter and, if the upper range is present, time shifting is not available beyond that time. If this value indicates an open range (one absolute start time for the timeshift recording), then the PSS server has not given any end-time for timeshift recording.

When both parameters are present, this indicates that the timeshift buffer is still being established. Recording has started at the lower bound of the interval and will progress until the specified depth is reached. Once the buffer depth is reached, then the timeshift buffering continues in a sliding window as described above and the PSS server uses the buffer header parameter for timeshift buffer reporting.

The ABNF syntax for the new headers is as follows:

```
3GPP-TS-CurrentRecording-Time='3GPP-TS-CurrentRecording-Time' COLON (npt-time-indication / utc-time-indication) CRLF

npt-time-indication = 'npt=' (npt-sec / npt-hhmmss)

utc-time-indication = 'clock=' utc-time

3GPP-TS-Buffer = '3GPP-TS-Buffer' COLON (depth / interval / interval_depth) [SEMIParameter-Ext] CRLF

depth = 'buffer-depth=' 1*DIGIT

interval = utc-range / npt-range

interval_depth = interval SEMI depth

Parameter-Ext =  (1*DIGIT ['.' 1*DIGIT]) / (1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7c / 0x7e))

COLON  =  < as defined in clause 5.5.4.2 >

SEMI     =  SWS ";" SWS ; semicolon;

SWS      = < as defined in clause 5.5.4.2>
```

Note: utc-range, npt-range, utc-time, npt-sec, and npt-hhmmss are defined in [5]RFC2326.

5.6.5 Timeshift buffer status updates

The GET_PARAMETER and the SET_PARAMETER methods allow the client and the server to synchronize timeshift buffer information.

The PSS client may query the PSS server at any time for the current timeshift buffer fill states. For this procedure, the PSS Client issues the RTSP GET_PARAMETER request message with the 3GPP-TS-Buffer and/or 3GPP-TS-CurrentRecording-Time tags in the message body. The MIME Type 'text/plain' is used for the RTSP message body.

The GET_PARAMETER response shall carry the 3GPP-TS-Buffer and the 3GPP-TS-CurrentRecording headers in the header fields and also in the RTSP response message body.

The server may update the current timshift information using the RTSP SET_PARAMETER method (S->C). The SET_PARAMETER request message shall carry the 3GPP-TS-Buffer and the 3GPP-TS-CurrentRecording headers in the header fields and also in the RTSP request message body. The PSS client shall confirm the reception with a '200 OK' RTSP response message or, if it client does not understand the parameters, it shall return '451 Invalid Parameters' and the PSS server shall not attempt further updates.

5.7 Support for Trick Mode Operations

PSS client and server may support trick mode operations such as fast/slow forward and rewind. The level of scale support may also depend on the selected streaming content.

A PSS Client may use the 'play.scale' feature tag to query for support of scaled playout. The 'play.scale' feature tag applies only to PSS servers and indicates the support for scale operations for media streaming.

If the server supports the scale value requested, and the content is capable of the scale value requested, the server shall serve the scaled stream to the client. If the server does not support the scale value requested or the content does not support the scale value requested then the server shall decide what level of scaled playout to serve to the client. When trick mode operations are supported, the 'Scale' header field of RTSP [5] shall be used for requesting trick mode operations.

Streaming with a scale value other than 1 should not change the streaming bitrate of the corresponding media stream.

The PSS Client is made aware of the scale values supported for the content via the 'X-Scale' media level attribute, or by using the 'scales' parameter in the GET_PARAMETER and SET_PARAMETER. The 'scales' parameter takes precedence in the case of conflicting values. The PSS server may temporarily omit the transmission of media data for one or more media streams on which playout with the selected scale is not possible. For example, this may mean that the video is scaled at a rate of 2, but the audio is omitted for the duration of the scaled playout as it is unable to be scaled.

A PLAY request with a 'Scale' indication that is not supported by any of the media streams of the streaming session may be ignored by the PSS server. The PSS server should use the closest supported scale value instead and it shall indicate the chosen value in the response, unless the selected scale value is 1.

The 'X-Scale' SDP attribute and the 'scales' parameter are defined according to the following ABNF syntax:

```
Scale="X-Scale:" payload_type SP scale_value *(";" scale_value) CRLF

scales="scales:" scale-spec *("," scale-spec) CRLF

scale-spec= stream-url "=" scale *(";" scale)

scale_value=["-"] 1*DIGIT [ "." *DIGIT]
```

In the SDP, the payload_type indicates the payload type of the media to which the scale values apply. The stream-url indicates the URL of the media stream to which the indicated scales are applicable. The scale_value is a decimal number that indicates the possible scale value that may be requested for the specified media stream. For a single media stream, multiple scale values are possible.

When timeshifting is used, the following applies:

- in the case that the PSS Client knows that, due to the PSS Server"s buffer, a particular request for scaled playback is impossible to complete, the PSS Client shall not request this scale value.

- in the case that the PSS Server is providing scaled playback and a buffer limit is reached, the PSS Server shall return the scale value of the stream to 1.

- the PSS client should take into account the playout scale when calculating its current position in the time shifting buffer, and take the time shifting buffer into account when requesting a playout scale.

6 Data transport

6.1 Packet based network interface

PSS clients and servers shall support an IP-based network interface for the transport of session control and media data. Control and media data are sent using TCP/IP [8] and UDP/IP [7]. An overview of the protocol stack can be found in figure 2 of the present document.

6.2 RTP over UDP/IP

6.2.1 General

The IETF RTP [9] provides means for sending real-time or streaming data over UDP (see [7]). The encoded media is encapsulated in the RTP packets with media specific RTP payload formats. RTP payload formats are defined by IETF. RTP also provides a protocol called RTCP (see clause 6 in [9]) for feedback about the transmission quality.

RTP/UDP/IP transport of speech, audio and video shall be supported. RTP/UDP/IP transport of timed text should be supported. Sending of RTCP shall be performed according to the used RTP profile, indicated RTCP bandwidth, and other RTCP related parameters. The transmission times of RTCP shall be controlled by algorithms performing as the ones specified in the RTP specification [9], and if AVPF is used according to [57]. For information on how the RTCP transmission interval depends on different values of the RTCP parameters, see Annex A.3.2.3.

6.2.2 RTP profiles

For RTP/UDP/IP transport of continuous media the following RTP profile shall be supported:

- RTP Profile for Audio and Video Conferences with Minimal Control [10], also called RTP/AVP;

For RTP/UDP/IP transport of continuous media the following RTP profile should be supported:

- Extended RTP Profile for RTCP-based Feedback (RTP/AVPF) [57], also called RTP/AVPF. A PSS client or server shall support the generic NACK message specified in section 6.2.1 of [57] if RTP retransmission is supported. A PSS client or server is not required to support the other feedback formats specified in section 6 of [57].

Clause A.3.2.3 in Annex A of the present document provides more information about the minimum RTCP transmission interval.

For integrity protected RTP/UDP/IP transport of continuous media, the following RTP profile should be supported:

- The Secure Real-time Transport Protocol (SRTP) [72], also called RTP/SAVP.

6.2.3 RTP and RTCP extensions

6.2.3.1 RTCP extended reports

A PSS client should implement the framework and SDP signalling of the RTP Control Protocol Extended Reports [58]. A PSS client should further implement the following report formats:

- Loss RLE Report Block defined in section 4.1of [58].

A PSS client should send the report block(s) indicated by SDP signalling from the PSS server. A PSS server may limit the report blocks size using SDP signalling. For best utility the client should report in every packet and provide redundancy by reporting also on past RTCP intervals. In cases where size restrictions prevent the client from both reporting on all the RTP packets and providing redundancy, the client shall stop the redundant reporting to address this restriction. If this action is still not enough to reduce the reports to satisfactory sizes, the client may then choose not to send the report in every packet.

6.2.3.2 RTCP App packet for client buffer feedback (NADU APP packet)

A PSS client supporting Signalling for Client Buffer Feedback (see clause 10.2.3) shall report the next application data unit to be decoded for buffer status reporting and rate adaptation by sending the RTCP APP packet. A NADU APP packet shall be sent only after the client has received at least one RTP packet on the media stream and shall be accompanied by a complementary RR packet. The RR and NADU packets shall contain information that represents a single simultaneous 'snapshot' of the media stream. The format of a generic RTCP APP packet is shown in Figure 3 below:

```
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 |   PT=APP=204  |             length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           SSRC/CSRC                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          name (ASCII)                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   application-dependent data                ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Generic Format of an RTCP APP packet.

For rate adaptation the name and subtype fields must be set to the following values:

name: The NADU APP data format is detected through the name "PSS0", i.e. 0x50535330 and the subtype.
subtype: This field shall be set to 0 for the NADU format.
length: The number of 32 bit words –1, as defined in RFC 3550 [9]. This means that the field will be 2+3*N, where N is the number of sources reported on. The length field will typically be 5, i.e. 24 bytes packets.
application-dependent data: One or more of the following data format blocks (as described in Figure 4) can be included in the application-dependent data location of the APP packet. The APP packets length field is used to detect how many blocks of data are present. The block shall be sent for the SSRCs for which there are a report block as part of either a Receiver Report or a Sender Report, included in the RTCP compound packet. A NADU APP packet shall not contain any other data format than the one described in figure 4 below.

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              SSRC                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Playout Delay            |             NSN               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Reserved           |   NUN   |    Free Buffer Space (FBS)    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: Data format block for NADU reporting

SSRC: The SSRC of the media stream the buffered packets belong to.

Playout delay (16 bits): The difference in milliseconds between the scheduled playout time of the next ADU to be decoded, (whose sequence number is indicated in the NSN field) and the current time when generating the RTCP packet that contains the NADU APP block, both measured on the media playout clock. The client shall always indicate this value, unless it is not well defined, when it may use the reserved value (0xFFFF). When the buffer is empty (the client has not yet received the packet with sequence number NSN), the playout delay is not well defined and the client should use the reserved value 0xFFFF for this field. When the media clock is not advancing (e.g. while paused or re-buffering), the playout delay corresponds to the difference between the playout time of the next ADU and the media time at which playout will resume.

The point at which the media playout clock is measured should be chosen such that, if the only packet in the buffer is that with sequence number NSN, the playout delay indicates the time remaining until the media playout will 'starve' and this stream might need re-buffering. In the calculations of playout delay above, this point is used to determine the playout point of a media packet even though actual playout may occur later in the decoding chain. The target buffer time (see clause 5.3.2.2) must be measured from the same point.
The playout delay allows the server to have a more precise value of the amount of time before the client will underflow. The playout delay shall be computed until the actual media playout (i.e., audio playback or video display).

NSN (16 bits): The RTP sequence number of the next ADU to be decoded for the SSRC reported on. In the case where the buffer does not contain any packets for this SSRC, the next not yet received sequence number shall be reported, i.e. an NSN value that is one larger than the least significant 16 bits of the RTCP SR or RR report block's "extended highest sequence number received".

NUN (5 bits): The unit number (within the RTP packet) of the next ADU to be decoded. The first unit in a packet has a unit number equal to zero. The unit number is incremented by one for each ADU in an RTP packet. In the case of an audio codec, an ADU is defined as an audio frame. In the case of H.264 (AVC), an ADU is defined as a NAL unit. In the case of H.263 and MPEG4 Visual Simple Profile, an ADU is defined as a whole or a part of an H.263 video picture or MPEG4 VOP that is included in a RTP packet. In the specific case of H.263, each packet carries a single ADU and the NUN field shall be thus set to zero. Future additions of media encoding or transports capable of having more than one ADU in each RTP payload shall define what shall be counted as an ADU for this format.

FBS (16 bit): The amount of free buffer space available in the client at the time of reporting. The reported free buffer space shall be less than or equal to the buffer space that has been reported as available for adaptation by the 3GPP-Adaptation RTSP header, see clause 5.3.2.2. The amount of free buffer space are reported in number of complete 64 byte blocks, thus allowing for up to 4194304 bytes to be reported as free. If more is available, it shall be reported as the maximal amount available, i.e. 4194304 with a field value 0xffff.

Reserved (11 bits): These bits are not used and shall be set to 0 and shall be ignored by the receiver.

6.2.3.3 RTP retransmission

6.2.3.3.1 General

A PSS client should implement RTP retransmission. A PSS server should implement RTP retransmission. A PSS client or server implementing RTP retransmission shall implement the payload format, SDP signalling and mechanisms of the

RTP retransmission payload format [81]. In addition to the specifications and recommendations in [81], a PSS client and server supporting RTP retransmission shall follow the definitions in the following clauses.

6.2.3.3.2 Multiplexing scheme

The RTP retransmission payload format [81] provides two different schemes for multiplexing the original and the retransmission stream, i.e. session-multiplexing and SSRC-multiplexing. PSS servers shall use SSRC-multiplexing and shall not use session-multiplexing.

6.2.3.3.3 RTCP retransmission request

PSS clients shall use the NACK feedback message format defined in the "Extended RTP Profile for RTCP-based Feedback (RTP/AVPF)" [57] for requesting the retransmission of RTP packets.

Before requesting the retransmission of RTP packets the client should assess whether a requested packet can be decoded in time by checking the latest receiver buffer status. If the client sends RTCP APP packets for client buffer feedback, as defined in section 6.2.3.2, the same assessment should be performed by the server, according to the latest RTCP APP packet it has received.

6.2.3.3.4 Congestion control and usage with rate adaptation

To avoid network congestion due to the additional bandwidth required for the retransmission of lost packets, a PSS server or client implementing RTP retransmission shall estimate the available link rate and adapt the total transmission rate of the RTP session, including retransmissions, to the available link rate. The actual algorithms providing link-rate estimation and transmission-rate adaptation are implementation specific. Rules and information sources for the estimation of the available link rate are described in clause 10.2.1 of the present document. To adapt the total transmission rate including retransmissions, a PSS server can e.g. skip retransmissions, use the transmission rate adaptation described in clause 10.2.2 of the present document or use any other suitable method.

If the server uses multiple streams for rate adaptation, the server may receive retransmission requests for a stream that is different from the one it is currently using. The server should thus not flush its retransmission buffer after switching streams.

6.2.4 RTP payload formats

For RTP/UDP/IP transport of continuous media the following RTP payload formats shall be used:

- AMR narrow-band speech codec (see clause 7.2) RTP payload format according to [11]. A PSS client is not required to support multi-channel sessions;
- AMR wideband speech codec (see clause 7.2) RTP payload format according to [11]. A PSS client is not required to support multi-channel sessions;
- Extended AMR-WB codec (see clause 7.3) RTP payload format according to [85];
- Enhanced aacPlus and MPEG-4 AAC codec (see clause 7.3) RTP payload format according to [13]; the size of audioMuxElements shall be limited to the maximum size of one audio frame, which is 6144 bits per AAC channel; moreover multiplexing of multiple audio frames into one audioMuxElement should be avoided if this would lead to fragmentation across RTP packets;
- MPEG-4 Visual video codec (see clause 7.4) RTP payload format according to RFC 3016 [13];
- H.263 video codec (see clause 7.4) RTP payload format according to RFC 4629 [14];
- H.264 (AVC) video codec (see clause 7.4) RTP payload format according to [92]. A PSS client is required to support all three packetization modes: single NAL unit mode, non-interleaved mode and interleaved mode. For the interleaved packetization mode, a PSS client shall support streams for which the value of the "sprop-deint-buf-req" MIME parameter is less than or equal to MaxCPB * 1000 / 8, inclusive, in which "MaxCPB" is the value for VCL parameters of the H.264 (AVC) profile and level in use, as specified in [90]. Parameter sets shall not be transmitted within the RTP payload, i.e., all parameter sets required for a session must be provided in the SDP;
- 3GPP timed text format (see clause 7.9) RTP payload format according to [80];

- DIMS (see subclause 8.3) RTP payload format according to [98];
- DRM encrypted RTP payload format according to clause K.1 in Annex K;
- RTP retransmission payload format according to [81].

NOTE: The payload format RFC 3016 for enhanced aacPlus and MPEG-4 AAC specify that the audio streams shall be formatted by the LATM (Low-overhead MPEG-4 Audio Transport Multiplex) tool [21]. It should be noted that the references for the LATM format in the RFC 3016 [13] point to an older version of the LATM format than included in [21]. In [21] a corrigendum to the LATM tool is included. This corrigendum includes changes to the LATM format making implementations using the corrigendum incompatible with implementations not using it. To avoid future interoperability problems, implementations of PSS client and servers supporting enhanced aacPlus and/or AAC shall follow the changes to the LATM format included in [21]. It should be noted further that the enhanced aacPlus signalling mode 'backwards compatible explicit signalling' (as defined in [21]) can not be used with LATM.

6.3 HTTP over TCP/IP

The IETF TCP provides reliable transport of data over IP networks, but with no delay guarantees. It is the preferred way for sending the scene description, text, bitmap graphics and still images. There is also need for an application protocol to control the transfer. The IETF HTTP [17] provides this functionality.

HTTP/TCP/IP transport shall be supported for:

- still images (see clause 7.5);
- bitmap graphics (see clause 7.6);
- synthetic audio (see clause 7.3A);
- vector graphics (see clause 7.7);
- text (see clause 7.8);
- timed text (see clause 7.9);
- SMIL scene description (see clause 8);
- presentation description (see clause 5.3.3).

HTTP/TCP/IP transport should be supported for:

- 3GP files for download and progressive download (see clause 7.10).

NOTE: DIMS scene description can be provided either over RTP/UDP/IP (see clauses 5.4 and 6.2.4) or contained in a 3GP file downloaded over HTTP/TCP/IP (see clauses 5.4 and this clause).

6.4 Transport of RTSP

Transport of RTSP shall be supported according to RFC 2326 [5].

7 Codecs

7.1 General

For PSS clients supporting a particular media type, corresponding media decoders are specified in the following clauses.

7.2 Speech

If speech is supported, the AMR decoder shall be supported for narrow-band speech [18][63][64][65]. The AMR wideband speech decoder, [20][66][67][68], shall be supported when wideband speech working at 16 kHz sampling frequency is supported.

7.3 Audio

If audio is supported, then one or both of the following two audio decoders should be supported:

- Enhanced aacPlus [86] [87] [88]
- Extended AMR-WB [82] [83] [84]

Specifically, based on the audio codec selection test results Extended AMR-WB is strong for the scenarios marked with blue, Enhanced aacPlus is strong for the scenarios marked with orange, and both are strong for the scenarios marked with green colour in the table below:

Content type Bit rate	Music	Speech over Music	Speech between Music	Speech
14 kbps mono				
18 kbps stereo				
24 kbps stereo				
24 kbps mono				
32 kbps stereo				
48 kbps stereo				

More recent information on the performance of the codecs based on more recent versions of the codecs can be found in TR 26.936 [95].

Enhanced aacPlus decoder is also able to decode AAC-LC content.

Extended AMR-WB decoder is also able to decode AMR-WB content.

In addition, MPEG-4 AAC Low Complexity (AAC-LC) and MPEG-4 AAC Long Term Prediction (AAC-LTP) object type decoders [21] may be supported. The maximum sampling rate to be supported by the decoder is 48 kHz. The channel configurations to be supported are mono (1/0) and stereo (2/0).

When a server offers an AAC-LC or AAC-LTP stream with the specified restrictions, it shall include the 'profile-level-id' and 'object' MIME parameters in the SDP 'a=fmtp' line. The following values shall be used:

Object Type	profile-level-id	object
AAC-LC	15	2
AAC-LTP	15	4

7.3a Synthetic audio

If a PSS client supports synthetic audio both the Scalable Polyphony MIDI (SP-MIDI) content format defined in Scalable Polyphony MIDI Specification [44] and the device requirements defined in Scalable Polyphony MIDI Device 5-to-24 Note Profile for 3GPP [45] should be supported.

SP-MIDI content is delivered in the structure specified in Standard MIDI Files 1.0 [46], either in format 0 or format 1.

In addition a PSS client supporting synthetic audio should also support both the Mobile DLS instrument format defined in [70] and the Mobile XMF content format defined in [71].

A PSS client supporting Mobile DLS shall meet the minimum device requirements defined in [70] in section 1.3 and the requirements for the common part of the synthesizer voice as defined in [70] in sections 1.2.1.2. If Mobile DLS is supported, wavetables encoded with the G.711 A-law codec (wFormatTag value 0x0006, as defined in [70]) shall also be supported. The optional group of processing blocks as defined in [70] may be supported. Mobile DLS resources are delivered either in the file format defined in [70], or within Mobile XMF as defined in [71]. For Mobile DLS files delivered outside of Mobile XMF, the loading application should unload Mobile DLS instruments so that the sound bank required by the SP-MIDI profile [45] is not persistently altered by temporary loadings of Mobile DLS files.

Content that pairs Mobile DLS and SP-MIDI resources is delivered in the structure specified in Mobile XMF [71]. As defined in [71], a Mobile XMF file shall contain one SP-MIDI SMF file and no more than one Mobile DLS file. PSS clients supporting Mobile XMF must not support any other resource types in the Mobile XMF file. Media handling behaviours for the SP-MIDI SMF and Mobile DLS resources contained within Mobile XMF are defined in [71].

7.4 Video

If a PSS client supports video, ITU-T Recommendation H.263 Profile 0 Level 45 decoder [22] shall be supported. In addition, a PSS client should support:

- H.263 Profile 3 Level 45 decoder [22];
- MPEG-4 Visual Simple Profile Level 3 decoder [24] with the following constraints:
    - The number of Visual Objects supported shall be limited to 1.
    - The maximum frame rate shall be 30 frames per second;
    - The maximum f_code shall be 2;
    - The intra_dc_vlc_threshold shall be 0;
    - The maximum horizontal luminance pixel resolution shall be 352 pels/line;
    - The maximum vertical luminance pixel resolution shall be 288 pels/VOP;
    - If AC prediction is used, the following restriction applies: QP value shall not be changed within a VOP (or within a video packet if video packets are used in a VOP). If AC prediction is not used, there are no restrictions to changing QP value;
- H.264 (AVC) Constrained Baseline Profile Level 1.3 decoder [90] without requirements on output timing conformance (Annex C of [90]).

If H.264 (AVC) High Profile is supported by a PSS client, the decoder shall support decoding any stream compliant to H.264 (AVC) High Profile Level 3.0 [43] with frame_mbs_only_flag=1, without requirements on output timing conformance (Annex C of ITU-T Recommendation H.264 [43]).

NOTE: H.264 (AVC) Main Profile is a subset of H.264 High Profile, and a High Profile decoder is required to be able to decode Main Profile streams.

The video buffer model given in Annex G of the present document should be supported if H.263 or MPEG-4 Visual is supported. It shall not be used with H.264 (AVC).

The H.264 (AVC) decoder in a PSS client shall start decoding immediately when it receives data (even if the stream does not start with an IDR access unit), or alternatively no later than it receives the next IDR access unit or the next recovery point SEI message, whichever is earlier in decoding order. Note that when the interleaved packetization mode of H.264 (AVC) is in use, de-interleaving is done normally before starting the decoding process. The decoding process for a stream not starting with an IDR access unit shall be the same as for a valid H.264 (AVC) bitstream. However, the client shall be aware that such a stream may contain references to pictures not available in the decoded picture buffer. The display behaviour of the client is out of scope of this specification.

A PSS client supporting H.264 (AVC) should ignore any VUI HRD parameters, buffering period SEI message, and picture timing SEI message in H.264 (AVC) streams or conveyed in the "sprop-parameter-sets" MIME/SDP parameter. Instead, a PSS client supporting H.264 (AVC) shall follow buffering parameters conveyed in SDP, as specified in clause 5.3.3.2, and in RTSP, as specified in clause 5.3.2.4. A PSS client supporting H.264 (AVC) shall also use the RTP timestamp or NALU-time (as specified in [92]) of a picture as its presentation time, and, when the interleaved RTP packetization mode is in use, follow the "sprop-interleaving-depth", "sprop-deint-buf-req", "sprop-init-buf-time", and "sprop-max-don-diff" MIME/SDP parameters for the de-interleaving process. However, if VUI HRD parameters, buffering period SEI messages, and picture timing SEI messages are present in the bitstream, their contents shall not contradict any of the parameters mentioned in the previous sentence.

NOTE: ITU-T Recommendation H.263 Profile 0 has been mandated to ensure that video-enabled PSS supports a minimum baseline video capability. Both H.263 and MPEG-4 Visual decoders can decode an H.263 Profile 0 bitstream. It is strongly recommended, though, that an H.263 Profile 0 bitstream is transported and stored as H.263 and not as MPEG-4 Visual (short header), as MPEG-4 Visual is not mandated by PSS.

7.5 Still images

If a PSS client supports still images, ISO/IEC JPEG [26] together with JFIF [27] decoders shall be supported. The support requirement for ISO/IEC JPEG only applies to the following two modes:

- baseline DCT, non-differential, Huffman coding, as defined in table B.1, symbol 'SOF0' in [26];
- progressive DCT, non-differential, Huffman coding, as defined in table B.1, symbol 'SOF2' [26].

7.6 Bitmap graphics

If a PSS client supports bitmap graphics, the following bitmap graphics decoders should be supported:

- GIF87a, [32];
- GIF89a, [33];
- PNG, [38].

7.7 Vector graphics

If a PSS client supports vector graphics, SVG Tiny 1.2 [42] [43] and ECMAScript [94] shall be supported.

NOTE 1: The compression format for SVG content is GZIP [59], in accordance with the SVG specification [42].

NOTE 2: Only codecs and MIME media types supported by PSS, as specified in clause 7 and in subclause 5.4, respectively, shall be used. In particular, PSS clients are not required to support the Ogg Vorbis format.

NOTE 3: Content creators of SVG Tiny 1.2 are strongly recommended to follow the content creation guidelines provided in Annex L.

NOTE 4: A DIMS client is capable of processing SVG Tiny 1.2 data.

7.8 Text

The text decoder is intended to enable formatted text in a SMIL presentation.

If a PSS client supports text it shall support

- text formatted according to XHTML Mobile Profile [47];
- rendering a SMIL presentation where text is referenced with the SMIL 2.0 "text" element together with the SMIL 2.0 "src" attribute.

If text is supported, the following character coding formats shall be supported:

- UTF-8, [30];
- UCS-2, [29].

NOTE: Since both SMIL and XHTML are XML based languages it would be possible to define a SMIL plus XHTML profile. In contrast to the presently defined SMIL Language Profile that only contain SMIL modules, such a profile would also contain XHTML modules. No combined SMIL and XHTML profile is specified for PSS. Rendering of such documents is out of the scope of the present document.

7.9 Timed text

PSS clients supporting timed text shall support [51]. Timed text may be transported over RTP or downloaded contained in 3GP files using Basic profile.

NOTE: A PSS client supporting timed text shall receive and parse 3GP files containing the text streams. This does not imply a requirement on PSS clients to be able to render other continuous media types contained in 3GP files, e.g. AMR and H.263, if such media types are included in a presentation together with timed text. Audio and video are instead streamed to the client using RTSP/RTP (see clause 6.2).

7.10 3GPP file format

3GP files [50] can be used by both PSS clients and PSS servers. The following profiles are used:

- Basic profile shall be supported by PSS clients if timed text is supported;
- Basic profile, Extended-presentation profile and Progressive-download profile should be supported by PSS clients;
- Streaming server profile should be supported by PSS servers.

7.11 Timed graphics

PSS clients supporting timed graphics shall support 3GPP TS 26.430 [109].

8 Scene description

8.1 General

There are several options for scene description in PSS:

- SMIL presentation, where the SMIL file is provided on its own or included as a primary item in a 3GP file (Extended-presentation profile);
- DIMS, where DIMS is included as a track in a 3GP file (Basic profile) or as a primary item (possibly in combination with a track) in a 3GP file (Extended-presentation profile).

The usage of SMIL and DIMS in 3GP files is defined in [50]. For pure RTSP/RTP-based streaming and 3GP files containing continuous media only, no separate scene description is required.

8.2 Synchronised Multimedia Integration Language

The 3GPP PSS uses a subset of SMIL 2.0 [31] as format of the scene description. PSS clients and servers with support for scene descriptions shall support the 3GPP SMIL Language Profile defined in [52]. This profile is a subset of the SMIL 2.0 Language Profile, but a superset of the SMIL 2.0 Basic Language Profile. Document [52] also includes an informative Annex A that provides guidelines for SMIL content authors.

NOTE: The interpretation of this is not that all streaming sessions are required to use SMIL. For some types of sessions, e.g. consisting of one single continuous media or two media synchronised by using RTP timestamps, SMIL may not be needed.

8.3 Dynamic and Interactive Multimedia Scenes

The 3GPP PSS uses DIMS [98] as a format of the scene description. PSS clients and servers with support for scene description shall support DIMS Mobile Profile at level 10 [98].

9 3GPP file format (interchange format for MMS)

The 3GPP file format is defined in [50].

10 Adaptation of continuous media

10.1 General

The PSS includes a number of protocols and functionalities that can be utilized to allow the PSS session to adapt transmission and content rates to the available network resources. The goal of this is of course to achieve highest possible quality of experience for the end-user with the available resources, while maintaining interrupt-free playback of the media. This requires that the available network resources are estimated and that transmission rates are adapted to the available network link rates. This can prevent overflowing network buffers and thereby avoid packet losses. The real-time properties of the transmitted media must be considered so that media does not arrive too late to be useful. This will require that media content rate is adapted to the transmission rate.

To avoid buffer overflows, resulting in that the client must discard useful data, while still allowing the server to deliver as much data as possible into the client buffer, a functionality for client buffer feedback is defined. This allows the server to closely monitor the buffering situation on the client side and to do what it is capable in order to avoid client buffer underflow. The client specifies how much buffer space the server can utilize and the minimum target level of protection the client perceives necessary to provide interrupt-free playback. Once this desired level of target protection is achieved, the server may utilize any resources beyond what is needed to maintain that protection level to increase the quality of the media or, the server may choose to leave the transmission rate alone and simply accrue additional time in the client buffer at the present rate. The server can also utilize the buffer feedback information to decide if the media quality needs to be lowered in order to avoid a buffer underflow and the resulting play-back interruption.

10.2 Bit-rate adaptation

The bit-rate adaptation for PSS is server centric in the meaning that transmission and content rate are controlled by the server. The server uses RTCP and RTSP as the basic information sources about the state of the client and network. This allows link-rate adaptation also when communicating with PSS clients of earlier releases, as long as they send RTCP receiver reports frequently enough.

10.2.1 Link-rate estimation

The actual algorithm providing the link-rate estimation is implementation specific. However, this chapter describes and gives rules for the different information sources that can be used for link-rate estimation.

10.2.1.1 Initial values

A PSS client should inform the server the quality of service parameters for the used wireless link. The known parameters should be included in the RTSP "3GPP-Link-Char" header (chapter 5.3.2.1) in either the RTSP SETUP or PLAY request. This enables the server to set some basic assumption about the possible bit-rates and link response. If the client has initially reported these parameters and they are changed during the session the client shall update these parameters by including the "3GPP-Link-Char" header in a SET_PARAMETER or OPTIONS request.

A PSS client should inform the server about initial bit-rate available over the link, if known. This reporting shall be done using the RTSP "Bandwidth" header in either the RTSP SETUP or PLAY request. The QoS negotiated guaranteed bit-rate is the best estimate for the bandwidth value.

10.2.1.2 Regular information sources

The basic information source giving regular reports useful for bit-rate estimations is the RTCP receiver reports as defined by [9]. The RTCP reporting interval is dependent on the RTP profile in use, the bit-rate assigned to RTCP, the average size of RTCP packets, and the number of reporting entities. Most of these parameters can be set or affected by the PSS server through signalling. This allows the server to configure the reporting interval to a desirable working point. See chapter 5.3.3.1 for specification on how the RTCP bandwidth is signalled by the server.

In most PSS RTP sessions the server and the client only have one SSRC each, thus providing the highest possible reporting rate. However some scenarios could result in that the number of used SSRC is larger, thereby possibly lowering the effective reporting interval for client, server or both.

The average size of the RTCP packets cannot be tightly controlled, but a loose control is possible by controlling which RTCP packet types that are used. This will depend on which of the below-listed RTCP extensions are in use.

The PSS server can signal the PSS client in SDP, to request that "Loss RLE Report Block" in RTCP XR (section 6.2.3) are used to report packet loss vectors.

10.2.2 Transmission adaptation

The transmission adaptation is implementation dependent. The 3GPP file format server extensions [50] provide a server the possibility to store alternative encodings useful for stream switching.

A server doing transmission rate adaptation through content rate adaptation shall still deliver content according to the SDP description of the media streams, e.g. a video stream delivered after content rate adaptation must still belong to the SDP announced profile and be consistent with any configuration. This will either put restrictions on the possible alternatives or require declaration of several RTP payload types or media encodings that might not be used.

10.2.3 Signalling for client buffer feedback

The client buffer feedback signalling functionality should be supported by PSS clients and PSS servers. For PSS clients and servers that support the client buffer feedback signalling functionality, the following parts shall be implemented:

- SDP service support, as described in clause 5.3.3.5.
- The size (in bytes) of the buffer the client provides for rate adaptation. It is signalled to the server through RTSP, as described in clause 5.3.2.2
- The target buffer protection time (in milliseconds). It is signalled to the server through RTSP, as described in clause 5.3.2.2.
- The client buffer status feedback information, including free buffer space, next ADU to be decoded and playout delay. It is signalled to the server via RTCP, as described in clause 6.2.3.2.

If a PSS server supports client buffer feedback, it shall include the attribute "3GPP-Adaptation-Support" in the SDP, as described in clause 5.3.3.5. If a PSS client supports client buffer feedback, upon reception of an SDP containing the "3GPP-Adaptation-Support" attribute, it shall include the "3GPP-Adaptation" header in the SETUP for each individual media. Furthermore, upon reception of a successful SETUP response (including "3GPP-Adaptation" header), the PSS client shall send NADU APP packets according to clause 5.3.3.5 and 6.2.3.2.

The "3GPP-Adaptation" header may be included in PLAY, OPTIONS and SET_PARAMETER requests in order to update the target buffer protection time value during a session. However, the target-protection-time is intended to be stable for the entire session with the server there are very few reasons for a client to modify the target buffer protection time once a session is established. The buffer size value shall not be modified during a session.

With the total buffer size, and the reported amount of free buffer space, the server can avoid overflowing the buffer. A server should assume that any sent RTP packet will consume receiver buffer space equal to the complete RTP packet size. For interleaved or aggregated media, the actual buffer space consumption may be slightly larger if buffering is done in the ADU domain. This is because each ADU may save metadata corresponding to the RTP header and payload fields, like timestamp and decoding sequence numbers individually. This should only be a problem if a server tries to fill exactly to the last free memory block.

The server can determine the time to underflow by calculating the amount of media time present in the buffer. This is done using the next ADU numbers, the highest received sequence number, and the playout delay, combined with the server's view of the sent ADUs and their decoding order and playout time. The information about the ADUs for 3GP files that are produced according to the streaming-server profile can be read from the "3gau" box [50]. It is also possible to derive some of the information about the ADUs from the media track, or hint-track, or the actual RTP packets.

A client needs to choose the target-time and the point on the playout timeline from which it will measure PlayoutDelay such that it will never re-buffer when the target-time is fulfilled. A client should typically begin rebuffering only when it has reached 0 ms buffered data. Once rebuffering has begun, the client should resume playback when the target-time has been fulfilled for all synchronized media streams.

The level of protection needed against transmission rate variations over a wireless network can be substantial (throughput variation because of network load, radio conditions, several seconds of interruption because of handovers, possible extra buffering to perform retransmission). In order to minimise the initial buffering delay, the client may choose an initial buffering that is less than the required buffering it has determined would be satisfactory. The client needs to take into account, however, that it may be unsafe to begin playback prior to fulfilling its target time. For this reason, the target buffer protection time indicates the amount of playable media (in time), which the client perceives necessary to have in its buffer. Therefore a server should not perform content adaptation towards higher content rates until the given target time of media units is available in the buffer.

It is important to note that target-protection-time is intended only to guide the server in its attempts to sustain or improve the quality of the media. There are many situations in which the target-protection time may not be respected by the server which will actually result in better media quality for the client (e.g. when the client sends a target-protection-time smaller than the perceived jitter or when the client sends a target-protection-time that is close to or exceeds the client buffer maximum). The only requirement the target-time places on the server is that the server shall not attempt to upshift prior to attaining the target-time.

Furthermore, while it is possible for the client to modify the target protection time in the 3GPP adaptation header with each RTSP request that is sent to the server, the target protection time is intended to be a stable value for the entire session with the server and should only be modified in circumstances where the client has a more accurate understanding of network and transmission jitter and the efficiency of its ability to process the network buffer. In these circumstances, adjusting the target time up could prevent buffer low points which will cause rebuffering or, adjusting the target time down could provide more head room to allow the server to adapt to the most appropriate rate.

10.3 Issues with deriving adaptation information (informative)

This clause attempts to provide some insight into the functions and issues that exist in deriving client"s buffer status in the server. The issues and the complexity of the functions depend on the media format, but can be characterised by media properties, in particular how much flexibility the media formats allows in transmission, decoding, and playout order. As there are three orderings of encoded media data that are possible, there are two re-orderings:

a) Data may be interleaved (i.e. the transmission order of data differs from the decoding order), and it must be de-interleaved before passing to the decoder.

b) There are forward references in the encoding, e.g. in a video stream, then those references are decoded 'early' (out of order) compared to playout order. Thus, the playout order in this case differs from the decode order.

In buffer management, we are trying to ensure

1. that the client's receiver buffer does not get over-filled (this is over-run);

2. that data does not arrive at an operation point after its need. Specifically, this means that ADUs should not be placed into the final playout queue with a timestamp that has already been passed in playout (this is under-run).

The parameters supplied enable a server to deduce at least this much. The server can always protect against buffer over-run by respecting the 'free space' that is periodically signalled by the client. This free-space is totalled over all data held before the decoder (decoder and de-interleave buffers). If the server desires more visibility, it can inspect the ADU that has been reported as 'next to decode'. If there has been no interleaving, the client holds all data between that ADU and the highest sequence number received, and will probably hold up to the last packet the server has sent. If interleaving is used, then there may have been ADUs that were sent after the reported ADU, but which passed out of both the de-interleaving and decoder buffers before that ADU. The server would have to analyze the de-interleaving process to work out which ADUs these are. The hint-track extension "3gau" to the 3GP file format [50] provides extended information about both the decoding and playout order in relation to transmission order of the ADUs. This extension does also provide the size of the ADUs to the server.

Protection against under-run is more subtle. It is in general not possible for the client to know which ADUs that are yet to be decoded (or yet to be received) that have earlier timestamps than ADUs already received and decoded. Therefore the client does not in fact know what is the 'latest playable timestamp', up to which it has received all the ADUs in the sequence to that time.

If the server does not adapt its transmission bit-rate and the transmission path has sufficient bit-rate, the parameters supplied at stream setup (such as the initial buffering delay) are sufficient to protect against under-run. The simple generalization of this is that if the server calculates its average bit-rate since starting the stream, and ensures that the average never falls below the bit-rate that would have been used without rate adaptation, it must be safe. Put in another way, the server may send a packet earlier than it would without rate-adaptation, but it might not be safe to send it later.

A more subtle analysis uses the reported information about the next-to-be-decoded ADU: the sequence number of the packet that contained it, the ADU number within that packet, and the offset (playout delay) of its timestamp (playback time) from the current playback time. Given the first pair of numbers, the server can find the ADU and therefore its timestamp. By subtracting the reported play-out delay from this timestamp, the server can now estimate the current playback time. It can find the earliest timestamp in the ADUs it has yet to transmit, and it can also examine the data that has been sent that will still be in the de-interleaving buffer, for the earliest timestamp still held in the client's de-interleaving buffer. If the earlier of these two timestamps is at, or close to, the current play time, the client has, or is about to, under-run.

Consider now the following cases, in increasing order of complexity:

1. simple data that is neither interleaved nor re-ordered for display (e.g. AMR without interleave, AAC, H.263, MPEG-4 video).

2. data that is interleaved, but not re-ordered (e.g. AMR with interleave).

3. data that is re-ordered, but not interleaved (AVC without interleave).

4. data that is both interleaved and re-ordered (AVC with interleave).

Consider now over-run and under-run protection for these streams. In all cases, the free-space can be used to protect against over-run, and the maintenance of the average rate at or above the static rate protects against under-run.

1. By subtracting the reported free-space from the overall buffer size (reported in stream setup) the buffered data can be calculated. If this is nearly exhausted, the buffer is about to under-run. However for codecs with variable bit-rate encoding, the buffered space may represent different amounts of playout time. In these cases the playout time present in the yet to be decoded part of the buffer can easily be calculated as the RTP timestamp difference between the latest ADU received by the client as reported implicitly by Highest Received Sequence number and the ADU reported by NADU.

2, The server can estimate the playback time as above. However to perform the calculation of the playout time of the buffer before the decoding, the server may need to maintain a list of the ADUs in the decoding order, rather than in transmission order. Also the data present in the de-interleaving buffer is not complete and would have holes in it and should not be considered to be playable. The server can determine, by looking at the decoding order of the different ADUs present in the transmitted packets, how far the client is expected to have a receiver buffer without holes, due to not yet transmitted packets.

3, In this case it may be fairly complicated to estimate the actual playout time of the un-decoded media. The reason is that the present RTP timestamp associated with the ADUs may fluctuate widely in ADUs consecutive in both transmission and decoding order, due to the early decoding of referenced ADUs. Therefore to perform an accurate estimation the server needs to make special consideration of any ADU with early decoding so that it does not skew the measurement. Note that for AVC bitstreams, a bound of the difference between presentation order and decoding order is given by the bitstream restriction parameter num_reorder_frames.

4, As 3 above, but with the further consideration of needing to perform any investigation in decoding order and consider the holes of the de-interleaving buffer.

11 Quality of Experience

11.1 General

The PSS Quality of Experience (QoE) metrics feature is optional for both PSS servers and clients, and shall not disturb the PSS service. A PSS server that supports the QoE metrics feature shall signal the activation and gathering of client QoE metrics when desired. QoE metrics can also be activated by a default setting via OMA-DM. A 3GPP PSS client supporting the feature shall perform the quality measurements in accordance to the measurement definitions, aggregate them into client QoE metrics and report the metrics to a server, which may or may not be the PSS server, using the QoE transport protocol when so requested. The way the QoE metrics are processed and made available is out of the scope of this specification.

11.2 QoE metrics

A PSS client should measure the metrics at the transport layer, but may also do it at the application layer for better accuracy.

The measurement period for the metrics is the period over which a set of metrics is calculated. The maximum value of the measurement period is negotiated via the QoE protocol as in clause 11.3. The measurement period shall not include any voluntary event that impacts the actual play, such as pause or rewind, or any buffering or freezes/gaps caused by them.

The following metrics shall be derived by the PSS client implementing QoE. All the metrics defined below are only applicable to at least one of audio, video, speech and timed text media types, and are not applicable to other media types such as synthetic audio, still images, bitmap graphics, vector graphics, and text. Any unknown metrics shall be ignored by the client and not included in any QoE report. Among the QoE metrics, corruption duration, successive loss of RTP packets, frame-rate deviation and jitter duration are of media level, whereas content switch time, initial buffering duration and rebuffering duration are of session level.

In the case of guaranteed delivery transports, such as HTTP as used in progressive download or HTTP-based streaming, metrics relating to loss or corruption (such as "Corruption duration", "Successive loss of RTP packets" and "Jitter duration") are not relevant and should be omitted from the report or report that no corruption has occurred.

11.2.1 Corruption duration metric

11.2.1.1 Default reporting format

Corruption duration, M, is the time period from the NPT time of the last good frame before the corruption (since the NPT time for the first corrupted frame cannot always be determined) or the start of the measurement period (whichever is later), to the NPT time of the first subsequent good frame or the end of the measurement period (whichever is sooner). A corrupted frame is either an entirely lost frame, or a media frame that has quality degradation and the decoded frame is not the same as in error-free decoding. A good frame is a "completely received" frame X that

- either it is a refresh frame (does not reference any previously decoded frames AND where none of the subsequently decoded frames reference any frames decoded prior to X);
- or does not reference any previously decoded frames;
- or only references previously decoded "good frames".

"Completely received" means that all the bits are received and no bit error has occurred.

Corruption duration, M, in milliseconds can be calculated according to the derivation of good frames as below:

a) A good frame can be derived by the client using the codec layer, in which case the codec layer signals the decoding of a good frame to the client. A good frame could also be derived by error tracking methods, but decoding quality evaluation methods shall not be used. An error tracking method may derive that a frame is a good frame even when it references previously decoded corrupted frames, as long as all the referenced pixels for generating the prediction signal were correctly reconstructed when decoding the reference frames. A decoding quality evaluation method may derive that a frame is a good frame even one or more pixels of the frame have not been correctly reconstructed, as long as the decoding quality is considered by the method as acceptable. Such a frame is not a good frame according to the definition above, which shall be strictly followed.

b) In the absence of information from the codec layer, a good frame is derived according to N, where N is optionally signalled from server to client and represents the maximum duration, in presentation time, between two subsequent refresh frames in milliseconds. After a corrupted frame, if all subsequent frames within N milliseconds in presentation time have been completely received, then the next frame is a good frame.

c) If N is not signalled, then it defaults to infinity (for video) or to one frame duration (for audio).

The optional parameter D is defined to indicate which of options a) and b) is in use. D is signalled from the client to the server. When D is equal to "a", option a) shall be in use, and the optional parameter T shall be present. When D is equal to "b", option b) shall be in use and the optional parameter T shall not be present.

The optional parameter N as defined in point b is used with the "Corruption_Duration" parameter in the "3GPP-QoE-Metrics" header. The optional parameter T is defined to indicate whether the client uses error tracking (when T is equal to "On") or not (when T is equal to "Off"). T is signalled from the client to the server.

The syntax for D, N and T to be included in the "Measure-Spec" (clause 5.3.2.3.1) is as follows:

D = "D" "=" "a" / "b"

N = "N" "=" 1*DIGIT

T = "T" "=" "On" / "Off"

The syntax for the "Metrics-Name Corruption_Duration" for the QoE-Feedback header is as defined in clause 5.3.2.3.2

The absence of an event is reported using the space (SP).

For the "Metrics-Name Corruption_Duration", the "Value" field in 5.3.2.3.2 indicates the corruption duration. The unit of this metrics is expressed in milliseconds. There is the possibility that corruption occurs more than once during a measurement period. In that case the value can occur more than once indicating the number of corruption events.

The value of "Timestamp" is equal to the NPT time of the last good frame inside the measurement period, in playback order, before the occurrence of the corruption, relative to the starting time of the measurement period. If there is no good frame inside the measurement period and before the corruption, the timestamp is set to the starting time of the measurement period.

11.2.1.2 XML reporting format

All the occurred corruption durations within each resolution period are summed and stored in the vector *TotalCorruptionDuration*. Within each resolution period the number of individual corruption events are summed up and stored in the vector *NumberOfCorruptionEvents.* These two vectors are then reported by the PSS client as Metric-Name "TotalCorruptionDuration" and "NumberOfCorruptionEvents" respectively. The use of error tracking is reported by setting the parameter *t* to "True" or "False".

11.2.2 Rebuffering duration metric

11.2.2.1 Default reporting format

Rebuffering is defined as any stall in playback time due to any involuntary event at the client side.

The syntax for the 'Metrics-Name Rebuffering_Duration" for the QoE-Feedback header is as defined in clause 5.3.2.3.2.

The absence of an event is reported using the space (SP).

For the "Metrics-Name Rebuffering_Duration", the "Value" field in 5.3.2.3.2 indicates the rebuffering duration. The unit of this metrics is expressed in seconds, and can be a fractional value. There is the possibility that rebuffering occurs more than once during a measurement period. In that case the metrics value can occur more than once indicating the number of rebuffering events.

The optional "Timestamp" indicates the time when the rebuffering has occurred since the beginning of the measurement period. The value of the "Timestamp" is equal to the NPT time of the last played frame inside the measurement period and before the occurrence of the rebuffering. If there is no played frame inside the measurement period, the timestamp is set to the starting time of the measurement period.

11.2.2.2 XML reporting format

All the occurred rebuffering durations are summed up over each resolution period of the stream and stored in the vector *TotalRebufferingDuration*. The number of individual rebuffering events for each resolution period are summed up and stored in the vector *NumberOfRebufferingEvents.* These two vectors are then reported by the PSS client as Metric-Name "TotalRebufferingDuration" and "NumberOfRebufferingEvents" respectively.

11.2.3 Initial buffering duration metric

11.2.3.1 Default reporting format

Initial buffering duration is the time from receiving the first media packet until playing starts.

The syntax for the "Metrics-Name Initial_Buffering_Duration" for the QoE-Feedback header is as defined in clause 5.3.2.3.2 with the exception that "Timestamp" in "Measure" is undefined for this metric. If the measurement period is shorter than the "Initial_Buffering_Duration" then the client should send this parameter for each measurement period as long as it observes it. The "Value" field indicates the initial buffering duration occurring during the current measurement period, where the unit of this metrics is expressed in seconds, and can be a fractional value. There can be only one "Measure" and it can only take one "Value". The absence of an event can be reported using the space (SP). "Initial_Buffering_Duration" is a session level parameter.

For instance, if the measurement period is set to one second, and the total initial buffering duration is 2.4 seconds, then the three first initial buffering duration values reported will be 1 second, 1 second and 0.4 seconds.

11.2.3.2 XML reporting format

The XML reporting format is identical to the default reporting format.

11.2.4 Successive loss of RTP packets

11.2.4.1 Default reporting format

This parameter indicates the number of RTP packets lost in succession per media channel.

The syntax for the "Metrics-Name Successive_Loss" for the QoE-Feedback header is as defined in clause 5.3.2.3.2.

The absence of an event can be reported using the space (SP).

For the "Metrics-Name Successive_Loss", the "Value" field indicates the number of RTP packets lost in succession. The unit of this metric is expressed as an integer equal to or larger than 1. There is the possibility that successive loss occurs more than once during a measurement period. In that case the metrics value can occur more than once indicating the number of successive losses.

The optional "Timestamp" indicates the time when the succession of lost packets has occurred. The value of the "Timestamp" is equal to the NPT time of the last received RTP packet inside the measurement period, in playback order, before the occurrence of the succession of lost packets, relative to the starting time of the measurement period. If there is no received RTP packet inside the measurement period and before the succession of loss, the timestamp is set to the starting time of the measurement period.

If a full run length encoding of RTP losses with sequence number information is desired, RTCP XR [RFC 3611] Loss RLE Reporting Blocks should be used instead of the successive loss metric.

11.2.4.2 XML reporting format

All the number of successively lost RTP packets are summed up over each resolution period of the stream and stored in the vector *TotalNumberofSuccessivePacketLoss*. The unit of this metric is expressed as an integer equal to or larger than 0. The number of individual successive packet loss events over each resolution period are summed up and stored in the vector *NumberOfSuccessiveLossEvents.* The number of received packets is also summed up over each resolution period and stored in the vector *NumberOfReceivedPackets.* These three vectors are reported by the PSS client as as Metric-Name "TotalNumberofSuccessivePacketLoss", "NumberOfSuccessiveLossEvents" and "NumberOfReceivedPackets" respectively.

11.2.5 Frame rate deviation

11.2.5.1 Default reporting format

Frame rate deviation indicates the playback frame rate information. Frame rate deviation happens when the actual playback frame rate during a measurement period is deviated from a pre-defined value.

The actual playback frame rate is equal to the number of frames played during the measurement period divided by the time duration, in seconds, of the measurement period.

The parameter FR that denotes the pre-defined frame rate value is used with the "Framerate_Deviation" parameter in the "3GPP-QoE-Metrics" header. The value of FR shall be set by the server. The syntax for FR to be included in the "Measure-Spec" (clause 5.3.2.3.1) is as follows:

FR = "FR" "=" 1*DIGIT "." 1*DIGIT

The syntax for the Metrics-Name "Framerate_Deviation" for the QoE-Feedback header is as defined in clause 5.3.2.3.2 with the exception that "Timestamp" in "Measure" is undefined for this metric. The absence of an event can be reported using the space (SP).

For the Metrics-Name "Framerate_Deviation", "Value" field indicates the frame rate deviation value that is equal to the pre-defined frame rate minus the actual playback frame rate. This metric is expressed in frames per second, and can be a fractional value, and can be negative. The metric value can occur only once for this metric.

11.2.5.2 XML reporting format

In the XML reporting format the frame rate is reported instead of the frame rate deviation. The metric "Framerate" indicates the average actual playback frame rate used during each resolution period. It is expressed in frames per second, and can be a fractional value. The average frame rate for each resolution period is stored in the vector *Framerate* and the vector is reported by the PSS client as Metric-Name "FrameRate".

11.2.6 Jitter duration

11.2.6.1 Default reporting format

Jitter happens when the absolute difference between the actual playback time and the expected playback time is larger than a pre-defined value, which is 100 milliseconds. The expected time of a frame is equal to the actual playback time of the last played frame plus the difference between the NPT time of the frame and the NPT time of the last played frame.

The syntax for the Metrics-Name "Jitter_Duration" for the QoE-Feedback header is as defined in clause 5.3.2.3.2.

The absence of an event can be reported using the space (SP).

For the Metrics-Name "Jitter_Duration", the "Value" field in 5.3.2.3.2 indicates the time duration of the playback jitter. The unit of this metrics is expressed in seconds, and can be a fractional value. There is the possibility that jitter occurs more than once during a measurement period. In that case the metric value can occur more than once indicating the number of jitter events.

The optional "Timestamp" field indicates the time when the jitter has occurred since the beginning of the measurement period. The value of the "Timestamp" is equal to the NPT time of the first played frame in the playback jitter, relative to the starting time of the measurement period.

11.2.6.2 XML reporting format

All the Jitter_Durations are summed up over each resolution period and stored in the vector *TotalJitterDuration*. The number of individual events over the resolution duration are summed up and stored in the vector *NumberOfJitterEvents.* These two vectors are then reported by the PSS client as Metric-Name "TotalJitterDuration" and "NumberOfJitterEvents" respectively.

11.2.7 Content Switch Time

11.2.7.1 Default reporting format

Fast content switching is defined in section 5.5 and allows for improving the switch time between different content accessible via the same RTSP server. Content switch time has a significant impact on the quality of experience for the user. The content switch time metric is used to report the time that elapses between the initiation of the content switch by the user and up to the time of reception of the first media packet from the new content or media stream.

The syntax for the metric 'Content_Switch_Time' for the QoE Feedback header is defined in clause 5.3.2.3.2.

The absence of a content switch event or the impossibility to determine the duration of a content switch can be reported using the space (SP).

For the metric name 'Content_Switch_Time', the 'Value' field in 5.3.2.3.2 indicates the duration of the content switch as defined above. The unit of this metric is expressed in milliseconds.

In case several content switch events have occurred during the measurement period, a list of values is reported each relating to the corresponding old content or media URL.

The optional 'Timestamp' field indicates the time when the content switch event was triggered by the user. The value of the 'Timestamp' is equal to the NPT time of the old content that corresponds to the content switch triggering time.

11.2.7.2 XML reporting format

All the Content_Switch_Times are summed up over each resolution period and stored in the vector *TotalContentSwitchTime*. The number of individual events over the resolution duration are summed up and stored in the vector *NumberOfContentSwitchEvents.* These two vectors are then reported by the PSS client as Metric-Name "TotalContentSwitchTime" and "NumberOfContentSwitchEvents" respectively.

11.2.8 Average Codec Bitrate

11.2.8.1 Default reporting format

The average codec bitrate is the bitrate used for coding 'active' media information during the measurement resolution period.

For audio media 'active' information is defined by frames containing audio. If the audio codec uses silence frames (SID-frames), these frames are not counted as "active", and the SID-frames and the corresponding DTX time periods are excluded from the calculation. Thus for audio media the average codec bitrate can be calculated as the number of audio bits received for 'active' frames , divided by the total time, in seconds, covered by these frames. The total time covered is calculated as the number of 'active' frames times the length of each audio frame.

For non-audio media the average codec bitrate is the total number of media bits played out during the measurement resolution period, divided by the length of the playout period. The playout period length is normally equal to the length of the measurement resolution period, but if rebuffering occurs the playout period will be shorter (i.e. any rebuffering time shall be ignored when calculating the codec bitrate).

The syntax for the metric "Average_Codec_Bitrate" is defined in sub-clause 5.3.2.3.2.

For the metric name 'Average_Codec_Bitrate', the 'Value' field in 5.3.2.3.2 indicates the codec bitrate as defined above. The unit of this metrics is expressed in kbit/s and can be a fractional value.

11.2.8.2 XML reporting format

The average codec bitrate value for each measurement resolution period shall be stored in the vector *AverageCodecBitrate*. The unit of this metrics is expressed in kbit/s and can be a fractional value. The vector is then reported by the PSS client as Metric-Name "AverageCodecBitrate".

11.2.9 Codec Information

11.2.9.1 Default reporting format

The codec information metrics contain details of the media codec used during the measurement period. The unit of this metric is a string value. No "white space" characters are allowed in the string values, and shall be removed if necessary.

For audio media the codec information contains the audio codec type, represented as in an SDP offer, for instance "AMR-WB/16000/1".

For video media, the codec information contains the video codec type, represented as in an SDP offer, for instance 'H263-2000/90000'. Furthermore, the video profile and level used, as well as the image size used shall be reported. For instance "profile=0;level=45" for the profile and level information and '176x144' for the image size. In some cases the profile and level is reported together, for instance "profile-level-id=42e00a". Note that the image size reported for each measurement resolution period shall be the one actually used, not the maximum size allowed by the SDP negotiation.

For timed text media, the codec information contains the text encoding, represented as in an SDP offer, for instance "3gpp-tt/1000".

The syntax for the metric "CodecInfo", 'CodecProfileLevel' and 'CodecImageSize' are defined in sub-clause 5.3.2.3.2.

There is the possibility that the codec information is changed during the measurement period. In that case the metrics can occur more than once indicating the codecs used.

The optional "Timestamp" field indicates the time when codec changes have occurred, relative to the beginning of the measurement period.

11.2.9.2 XML reporting format

The codec info, profile/level and codec image size value for each measurement resolution period shall be stored in the vectors C*odecInfo*, *CodecProfileLevel* and *CodecImageSize* respectively. If the metric values in these vectors for a measurement resolution period are unchanged from the previous values in the respective vector, it is allowed to put the value '=' in the vector to indicate this. These three vectors are reported by the PSS client as as Metric-Name "CodecInfo", "CodecProfileLevel" and "CodecImageSize" respectively.

11.2.10 Buffer Status

11.2.10.1 Default reporting format

The buffer depth is the number of seconds of future media which resides in the buffer. It is calculated as the difference between the latest playout time of the media units in the buffer minus the current playout time. If the calculation result is negative, the buffer depth shall be set to zero. The buffer depth metric shall be calculated close to the end of each measurement period.

The unit of the metric bufferDepth is in seconds and can be a fractional value. If the length of the media is known, and if all remaining media already is buffered, then the boolean metric allContentBuffered shall be set to true.

The syntax for the metric "bufferDepth" and "allContentBuffered" is defined in sub-clause 5.3.2.3.2.

11.2.10.2 XML reporting format

The buffer depth close to the end of each measurement period shall be stored in the vector *bufferDepth*. The vector and the *allContentBuffered* status is then reported at the end of each reporting period.

11.3 The QoE protocol

11.3.1 General

PSS clients and servers supporting QoE Metrics shall support the QoE protocol described below.

The RTSP and SDP based protocol extensions (see clauses 5.3.2.3 and 5.3.3.6) are used for transport and negotiation of the QoE metrics between the PSS client and the PSS server. As an alternative, OMA-DM and HTTP can also be used for QoE configuration and reporting (see clauses 5.3.3.8 and 5.3.2.3.3).

The QoE metrics negotiation starts with the response to the DESCRIBE request, if the metrics information is embedded in the SDP data (as described in example 1 in clause 11.3.2). For the case of locally stored SDP which contains QoE-Metrics attribute, the negotiation starts with client"s SETUP request. If the PSS client supports QoE metrics, then it shall send a SETUP request containing the selected (i.e. accepted by client)/modified (for re-negotiation) QoE metrics for either session level, or the media level, which is being set-up. Such a SETUP request is shown in example 2 in clause 11.3.3.

Upon receiving this SETUP request, the server shall return the RTSP Response with the "accepted" QoE metrics (i.e. metrics and metrics values which are identical to the ones in the client"s request and accepted by the server) and the "re-negotiation" QoE metrics (i.e. metrics and metrics values which are not identical to the ones in the client"s request and modified for re-negotiation by the server) .The echoing of the "accepted" QoE metrics is for re-acknowledging the client. The server may also reject the changes made by the client, i.e. reject the "re-negotiation" QoE metrics. If the server rejects the changes, it shall either set new values or resend the modified metrics back to the client, or it shall ignore the "re-negotiation" metrics and not re-acknowledge them. Any QoE metric that has been acknowledged as "accepted" by the server shall not be re-negotiated, i.e., it shall not be resent in the "3GPP-QoE-Metrics" header in the next RTSP request and shall not be re-acknowledged in the next RTSP response.

If the server does not approve the modifications done by the client, they should continue to re-negotiate. However, negotiations shall not exceed 4 round trips, in order to bound the negotiation process. It must be noted that each time the "QoE-Metrics" header field is sent in an RTSP request, it shall also be present in the response corresponding to that particular request. Otherwise, the receiver of the response shall assume that the other end does NOT support QoE metrics.

If there is no DESCRIBE – RTSP Response pair sending at the beginning of the RTSP signalling (see Figure 11.2), it means that the SDP description is received by other means. If such an SDP contains the "3GPP-QoE-Metrics" attribute, the negotiation happens in the same way as it is described above, i.e. starts with SETUP request containing "3GPP-QoE-Metrics" header. If the SDP does not contain the "3GPP-QoE-Metrics" attribute and the server would still like to check whether the client supports QoE Protocol or not, the server shall include the "3GPP-QoE-Metrics" header containing the initial QoE metrics in the SETUP response. If the PSS client sends the QoE metrics information in the next request (indicating that it supports QoE Protocol), the negotiation shall continue until the mutual agreement is reached or the negotiation limit is reached. If pipelined startup is not in use and the client does not send QoE metrics information in the next request to SETUP response, then the server shall assume that the client does not support QoE metrics. In case pipelined startup is in use, the server may initiate QoE negotiation but it should not expect an answer from the PSS client.

In case of switching without the SDP, the PSS client shall assume that the same QoE metrics as negotiated for the old stream will be used for the new stream. In the PLAY response, the server includes the '3GPP-QoE-Metrics' header to acknowledge the QoE metric mapping to the new media streams or to change them.

During fast content switching with SDP, the client shall indicate the QoE metrics to be used for the new content using the '3GPP-QoE-Metrics' header. The client should use the already negotiated parameters as much as possible to avoid further negotiations. The server shall either acknowledge the proposed QoE metrics or continue negotiation beyond the PLAY response message. It is possible to turn off the metrics during a streaming session. In clause 11.3 an example of messages, where the metrics are set to "Off" is given. The metrics can be set to "Off" at session level or at media level. The request url indicates what level is used. If no url is used, then 'Off' applies to session level. The server should use OPTIONS (with Session ID) or SET_PARAMETER RTSP methods to turn off the QoE feedback.

A client should not send QoE feedback during RTSP ready state. After the ready state is ended (i.e., RTSP state=playing), the periodic feedback and normal operations continue. This reduces the network load in the uplink and downlink directions, and the processing overhead for the PSS client. When an RTSP PLAY request is sent by the PSS client after a PAUSE, the clock for the measurement period (based on the defined "Sending Rate") shall be reset.

If there are multiple non-aggregated sessions, i.e. each media delivery is initiated by a different PLAY request, the QoE metrics are negotiated and reported for each session separately.

All the QoE Metrics in the following examples are fictitious. Clause 11.2 defines the actual QoE Metrics.

11.3.2 Metrics initiation with SDP

QoE metrics initiation with SDP shall be done according to clause 5.3.3.6.

This following example shows the syntax of the SDP attribute for QoE metrics. The session level QoE metrics description (Initial buffering duration and rebufferings) are to be monitored and reported only once at the end of the session. Also video specific description of metrics (corruptions and decoded bytes) is to be monitored and reported every 15 seconds from the beginning of the stream until the time 40s. Finally, audio specific description of metrics (corruptions) is to be monitored and reported every 20 seconds from the beginning until the end of the stream.

EXAMPLE 1:

```
S->C   RTSP/1.0 200 OK
Cseq: 1
Content-Type: application/sdp
Content-Base: rtsp://example.com/foo/bar/baz.3gp/
Content-Length: 800
Server: PSSR7 Server

v=0
o=- 3268077682 433392265 IN IP4 63.108.142.6
s=QoE Enables Session Description Example
e=support@foo.com
c=IN IP4 0.0.0.0
t=0 0
a=range:npt=0-83.660000
a=3GPP-QoE-Metrics:metrics={Initial_Buffering_Duration|Rebuffering_Duration};rate=End
a=control:*
m=video 0 RTP/AVP 96
b=AS:28
a=3GPP-QoE-Metrics:metrics={Corruption_Duration|Decoded_Bytes};rate=15;range:npt=0-40
a=control:trackID=3
a=rtpmap:96 MP4V-ES/1000
a=range:npt=0-83.666000
a=fmtp:96profile-level-id=8;config=000001b008000001b50900012000
m=audio 0 RTP/AVP 98
b=AS:13
a=3GPP-QoE-Metrics:metrics={Corruption_Duration};rate=20
a=control:trackID=5
a=rtpmap:98 AMR/8000
a=range:npt=0-83.660000
a=fmtp:98 octet-align=1
a=maxptime:200
```

11.3.3 Metrics initiation/termination with RTSP

QoE Metrics initiation with RTSP can be done according to clause 5.3.2.3.1

In the following example it is shown how to negotiate QoE metrics during RTSP session setup.

EXAMPLE 1 (QoE metrics negotiation):

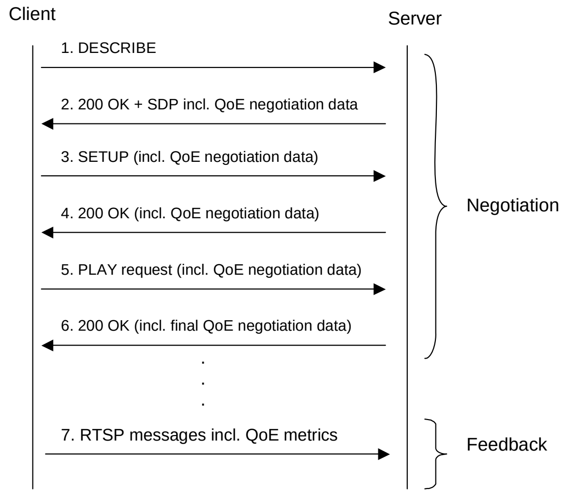


Figure 11.1: QoE metrics negotiation

C->S SETUP rtsp://example.com/foo/bar/baz.3gp/trackID=3 RTSP/1.0
Cseq: 2
3GPP-QoE-Metrics:url='rtsp://example.com/foo/bar/baz.3gp/trackID=3';
metrics={Corruption_Duration|Decoded_Bytes};rate=10; Range:npt=0-40,
url='rtsp://example.com/foo/bar/baz.3gp';
metrics={Initial_Buffering_Duration|Rebuffering_Duration};rate=End

In the above SETUP request, the client modifies the sending rate of the QoE metrics for the control URL 'rtsp://example.com/foo/bar/baz.3gp/trackID=3' from 15 to 10 (compared to the initial SDP description).

Assuming that the server acknowledged the changes, the server will send back a SETUP response as follows:

S->C RTSP/1.0 200 OK
Cseq: 2
Session: 17903320
Transport: RTP/AVP;unicast;client_port=7000-7001;server_port= 6970-6971
3GPP-QoE-Metrics:url='rtsp://example.com/foo/bar/baz.3gp/trackID=3';
metrics={Corruption_Duration|Decoded_Bytes};rate=10;Range:npt=0-40,
url='rtsp://example.com/foo/bar/baz.3gp';
metrics={Initial_Buffering_Duration|Rebuffering_Duration};rate=End

EXAMPLE 2 (QoE metrics negotiation – no DESCRIBE – 200/OK):

An example is shown in Figure 11.2 and can make use of the same RTSP header defined in clause 5.3.2.3.

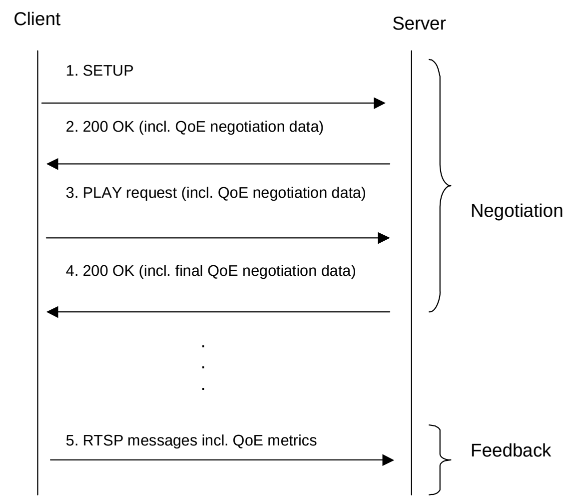


Figure 11.2: QoE metrics negotiation (no DESCRIBE-200/OK)

EXAMPLE 3 (QoE metrics negotiation – fast content switching with the SDP)

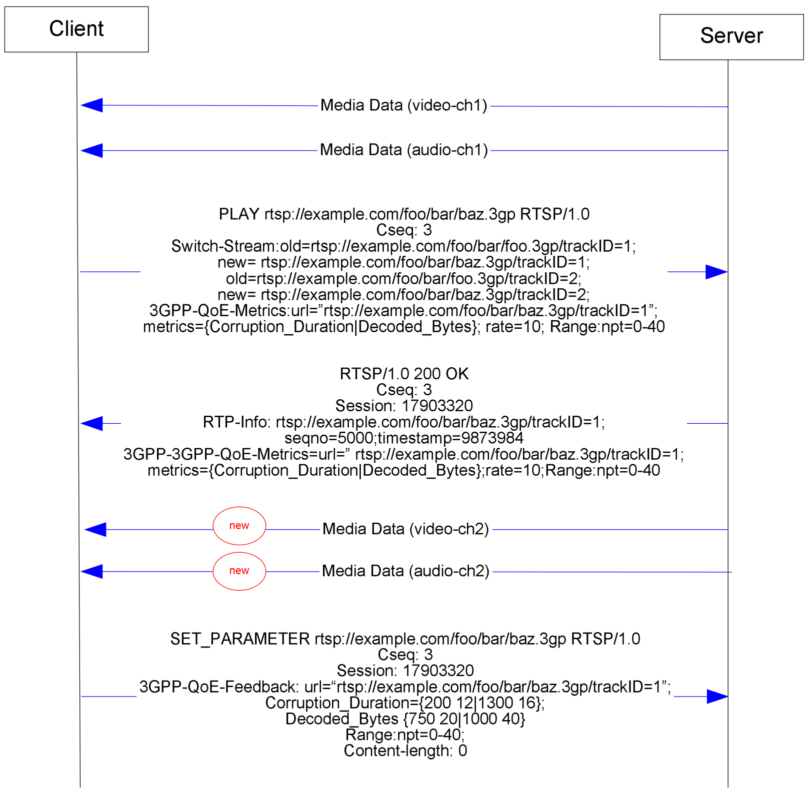


Figure 11.3: QoE metrics negotiation during content switch with SDP

C->S PLAY rtsp://example.com/foo/bar/baz.3gp RTSP/1.0
Cseq: 3
Session: 17903320
Switch-Stream:old='rtsp://example.com/foo/bar/foo.3gp/trackID=1';new='
rtsp://example.com/foo/bar/baz.3gp/trackID=1';
old='rtsp://example.com/foo/bar/foo.3gp/trackID=2';new='
rtsp://example.com/foo/bar/baz.3gp/trackID=2';
3GPP-QoE-Metrics:url='rtsp://example.com/foo/bar/baz.3gp/trackID=1';
metrics={Corruption_Duration|Decoded_Bytes};rate=10; Range:npt=0-40

In the above PLAY request, the client reuses the already negotiated sending rate of the QoE metrics for the control URL 'rtsp://example.com/foo/bar/baz.3gp/trackID=1' (compared to the above indicated range in the SDP description, which is 15).

Assuming that the server acknowledged the changes, the server will send back a PLAY response as follows:

S->C RTSP/1.0 200 OK
Cseq: 3
Session: 17903320
RTP-Info: rtsp://example.com/foo/bar/baz.3gp/trackID=1;seq=5000;rtptime=9873984 3GPP-
3GPP-QoE-Metrics=url=' rtsp://example.com/foo/bar/baz.3gp/trackID=1;
metrics={Corruption_Duration|Decoded_Bytes};rate=10;Range:npt=0-40

EXAMPLE 4 (QoE metrics negotiation – fast content switching without the SDP)

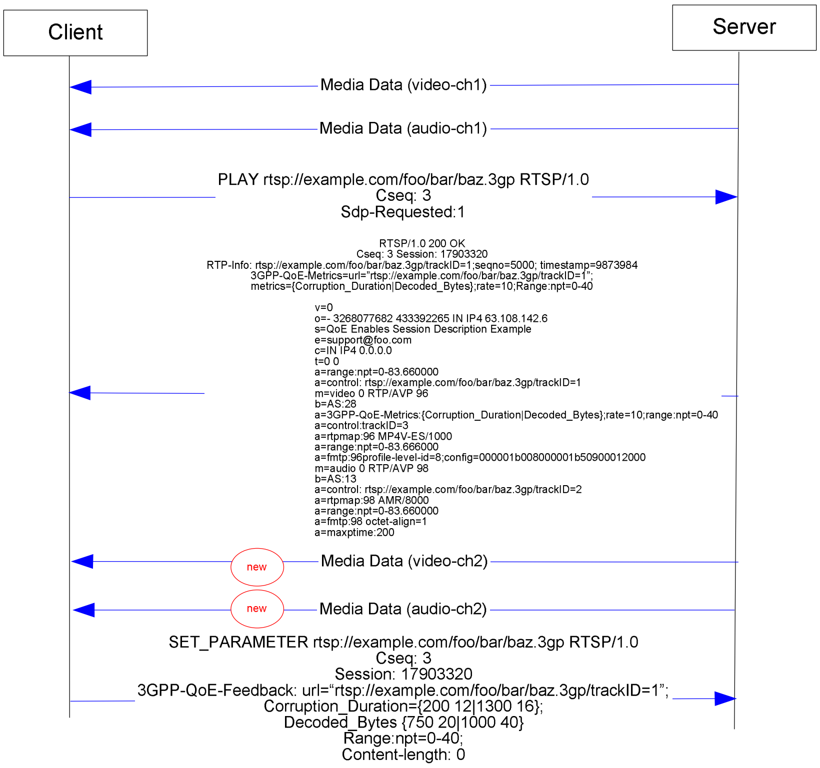


Figure 11.4: QoE metrics negotiation during content switch with SDP

C->S PLAY rtsp://example.com/foo/bar/baz.3gp RTSP/1.0
Cseq: 3
Session: 17903320
Sdp-Requested:1

In the above PLAY request, the client switches to new content without having the SDP. By consequence, the client does not have any information about the requested QoE parameters and cannot indicate those in the PLAY request.

In the case of a successful switch, the server indicates the QoE metrics in the SDP and in the 3GPP-QoE-Metrics header. The server reuses the already negotiated feedback period of 10 seconds.

S->C RTSP/1.0 200 OK
Cseq: 3
Session: 17903320
RTP-Info: rtsp://example.com/foo/bar/baz.3gp/trackID=1;seq=5000;rtptime=9873984 3GPP-
3GPP-QoE-Metrics=url='rtsp://example.com/foo/bar/baz.3gp/trackID=1';
metrics={Corruption_Duration|Decoded_Bytes};rate=10;Range:npt=0-40

```
v=0
o=- 3268077682 433392265 IN IP4 63.108.142.6
s=QoE Enables Session Description Example
e=support@foo.com
c=IN IP4 0.0.0.0
t=0 0
a=range:npt=0-83.660000
a=control: rtsp://example.com/foo/bar/baz.3gp/trackID=1
m=video 0 RTP/AVP 96
b=AS:28
a=3GPP-QoE-Metrics:{Corruption_Duration|Decoded_Bytes};rate=10;range:npt=0-40
a=control:trackID=3
a=rtpmap:96 MP4V-ES/1000
a=range:npt=0-83.666000
a=fmtp:96profile-level-id=8;config=000001b008000001b50900012000
m=audio 0 RTP/AVP 98
b=AS:13
a=control: rtsp://example.com/foo/bar/baz.3gp/trackID=2
a=rtpmap:98 AMR/8000
a=range:npt=0-83.660000
a=fmtp:98 octet-align=1
a=maxptime:200
```

The client may further negotiate the offered QoE metrics using OPTIONS or SET_PARAMETER methods.

EXAMPLE 4 (setting the metrics off):

In this example, the metrics are switched off at session level (for all media).

```
C->S, S->C    SET_PARAMETER rtsp://example.com/foo/bar/baz.3gp RTSP/1.0
              Cseq: 302
              Session: 17903320
              3GPP-QoE-Metrics: Off
              Content-length: 0
```

The response for setting the metrics off would be:

```
S->C, C->S    RTSP/1.0 200 OK
              Cseq: 302
              Session: 17903320
              3GPP-QoE-Metrics: Off
```

11.3.4 Sending the metrics feedback with RTSP

QoE Metric feedback with RTSP can be formatted and sent according to clause 5.3.2.3.2.

The following example shows that during the monitoring time 2 corruption periods have occurred. Each value indicates the duration (in milliseconds) of each corruption period.

EXAMPLE 5 (Feedback):

```
C->S          SET_PARAMETER rtsp://example.com/foo/bar/baz.3gp RTSP/1.0
              Cseq: 302
              Session: 17903320
              3GPP-QoE-Feedback:
              url='rtsp://example.com/foo/bar/baz.3gp/trackID=3';Corruption_Duration={200 1300}
              Content-length: 0
```

The following example shows that during the monitoring time 2 corruption periods have occurred. Each values couple indicates the duration (in milliseconds) of each corruption period and the timestamp of the corruption (for example, the first corruption occurred at second 12 and lasted 200 milliseconds).

EXAMPLE 6 (Feedback with timestamps and range):

C->S SET_PARAMETER rtsp://example.com/foo/bar/baz.3gp RTSP/1.0
Cseq: 302
Session: 17903320
3GPP-QoE-Feedback: url='rtsp://example.com/foo/bar/baz.3gp/trackID=1';
Corruption_Duration={200 12|1300 16};Range:npt=10-20;Content_Switch_Time={1055 34498}
Content-length: 0

In the following example there are no events to report.

EXAMPLE 7 (Feedback with no events):

C->S SET_PARAMETER rtsp://example.com/foo/bar/baz.3gp RTSP/1.0
Cseq: 302
Session: 17903320
3GPP-QoE-Feedback: url='rtsp://example.com/foo/bar/baz.3gp/trackID=3';Corruption_Duration={
}
Content-length: 0

12 Adaptive HTTP Streaming

12.1 System Description

The 3GPP Adaptive HTTP-Streaming protocol provides a streaming service. This enables delivering content from standard HTTP servers to an HTTP-Streaming client and enables caching content by standard HTTP caches.

Figure 12.1 shows the architecture for Adaptive HTTP streaming. This specification only deals with the specification of interface 1 between the HTTP-Streaming Client and the HTTP-Streaming Server. All other interfaces are out-of-scope of this specification.

It is assumed that the HTTP-Streaming Client has access to a Media Presentation Description (MPD). An MPD provides sufficient information for the HTTP-Streaming Client to provide a streaming service to the user by sequentially downloading media data from an HTTP server and rendering the included media appropriately.

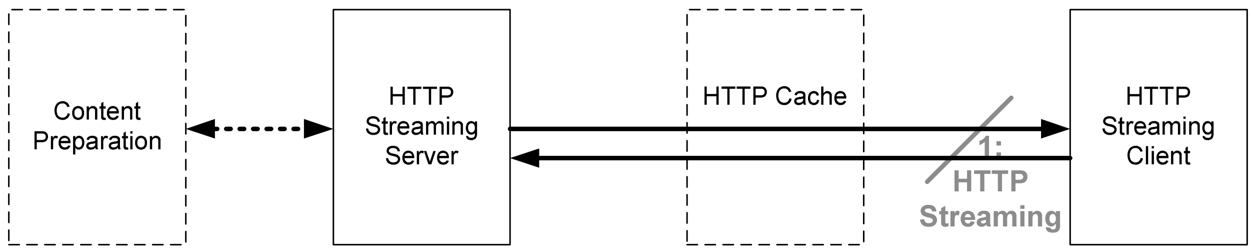


Figure 12.1 System Architecture for Adaptive HTTP Streaming

To initiate the streaming service to the user, the HTTP Streaming Client establishes a Media Presentation by downloading the relevant metadata and subsequently the media data. The Media Presentation is defined in clause 12.2. The protocols used in this specification are specified in clause 12.3. The usage of 3GP file format as media container format is specified in clause 12.4. The codecs are introduced in clause 12.5. Guidelines on the Client Behaviour are presented in clause 12.6. Security-related aspects are addressed in clause 12.7.

12.2 Media Presentation

12.2.1 Introduction

A Media Presentation is a structured collection of data that is accessible to the HTTP-Streaming Client. The HTTP-Streaming client requests and downloads media data information to present the streaming service to the user.

A Media Presentation is described in a Media Presentation Description (MPD) including any possible updates of the MPD. The MPD, including all attributes and elements, is specified in clause 12.2.5.

A Media Presentation fconsists of

- A sequence of one or more Periods.
- Each Period contains one or more Representations from the same media content.
- Each Representation consists of one or more Segments.
- Segments contain media data and/or metadata to decode and present the included media content.

The Media Presentation has a time line that is defined by the concatenation of the timeline of each Period. The timeline in each Period is common to all Representations.

The Media Presentation may be of type OnDemand or Live. The MPD attribute type provides the type of the Media Presentation.

If present, the MPD attribute *availabilityStartTime* gives the earliest time at which the Media Presentation is available at the server.

If present, the MPD attribute *availabilityEndTime* gives the time after which the Media Presentation will no longer be available at the server.

If present, the MPD attribute *mediaPresentationDuration* describes the duration of the entire Media Presentation. If not present, the duration of the Media Presentation is unknown.

12.2.2 Period

A Media Presentation consists of one or more Periods. Periods are defined by *Period* elements in the MPD.

Each Period has an attribute *start*.

For live services, the sum of the *start* attribute of the Period and the MPD attribute *availableStartTime* specifies the availability time of the Period in UTC format, in particular the first Media Segment of each Representation in this Period.

For on-demand services the *start* attribute of the first Period shall be 0. For any other Period the *start* attribute specifies the timeoffset between the start time of the Period relative to the *start* time of the first Period.

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

Period start times are precise. They reflect the actual timing resulting from playing the media of all prior Periods.

12.2.3 Representation

Each Period consists of one or more *Representations*.

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

A Representation starts at the *start* of the Period and continues to the end of the Period.

A Representation consists of one or more Segments.

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

Each Representation includes one or more media components, where each media component is an encoded version of one individual media type such as audio, video or timed text. Media *components* are time-continuous across boundaries of consecutive Media Segments within one Representation.

Representations are assigned to a group indicated by the *group* attribute. Representations in the same group are alternatives to each other. The media content within one Period is sufficiently represented by:

- either one Representation from group 0, if present,
- or the combination of at most one Representation from each non-zero group.

The timing within each Representation is relative to the start time of the Period that contains this Representation.

12.2.4 Segments

12.2.4.1 Definition

A Segment is defined as a unit that can be uniquely referenced by an http-URL included in the MPD, where an http-URL is defined as an <absolute-URI> according to RFC3986 [60], clause 4.3, with a fixed scheme of 'http://' or https://, possibly restricted by a byte range if the *range* attribute is provided. The byte range is expressed as a *Ranges*-specifier as defined in RFC2616 [17], section 14.35.1 restricted to a single expression identifying a contiguous range of bytes.

The Initialisation Segment contains initialisation information for accessing the Representation. The Initialisation Segment shall not contain any media data.

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

- is assigned a unique MPD URL Element.
- is explicitly or implicitly assigned a start time relative to the start of the Representation provided by the MPD. The client can therefore download the appropriate Segments in regular play-out mode or after seeking. The start time shall be drift-free between the time indicated in the MPD and internal clock of the Media Segments, i.e. the accuracy of the start time documented in the MPD relative to the internal clock does not depend on the position of the Segment in the Representation.
- provides random access information, namely whether this Representation can be randomly accessed within this Segment and if yes, how to randomly access the Media Presentation within this Segment, e.g. exact timing, byte position. There is no requirement that a Media Segment starts with a random access point (RAP), but it is possible to signal in the MPD that all Segments within a Representation start with a RAP. The first Media Segment of a Representation shall always start with a RAP.
- ´ when considered in conjunction with the information and structure of the MPD, contains sufficient information to time-accurately present each contained media component in the Representation without accessing any previous Media Segment in this Representation provided that the Media Segment contains a RAP. The time-accuracy enables seamlessly switching Representations and jointly presenting multiple Representations.
- may contain information for randomly accessing subsets of the Segment by using partial HTTP GET requests.

12.2.4.2 Segment URLs and Media Segment Start Times

12.2.4.2.1 Overview

Each Representation contains exactly one *SegmentInfo* Element that together with a possibly present *SegmentInfoDefault* Element on Period level permits to generate the Segment access information of each Segment within a Representation.

Specifically, the combination of the *SegmentInfo* Element and the *SegmentInfoDefault* Element contains sufficient information to generate a list of Media Segment URLs (possibly restricted by byte ranges) and Media Segment start times relative to the start of the Representation, for example by using the procedures described in Section 12.6.3.

The following rules apply for an MPD:

- URLs within the MPD may be relative or absolute as defined in RFC 3986 [60]. Relative URLs at each level of the MPD are resolved with respect to the *baseURL* attribute specified at that level of the document or the document 'base URI' as defined in RFC3986 Section 5.1 in the case of the *baseURL* attribute at the MPD level.. If only relative URLs are specified and the document base URI cannot be established according to RFC3986 then the MPD cannot be interpreted. *SegmentInfo* elements may contain at most one *InitialisationSegmentURL* element. If no *InitialisationSegmentURL* element is present in a *SegmentInfo* Element element, then each Media Segment within the Representation shall be self-initialising.

- *SegmentInfo* elements may contain at most one *InitialisationSegmentURL* element. If no *InitialisationSegmentURL* element is present in a *SegmentInfo* element, then each Media Segment within the Representation shall be self-initialising.
- The elements *SegmentInfoDefault* and *SegmentInfo* as well as some other timing attributes in the MPD define the URLs (possibly restricted by byte ranges) and approximate time spans within a Period of the available Media Segments. An example segment list generation process as specified in section 12.6.3 generates a valid list of Media Segments.
- Each *SegmentInfo* element shall contain either one *UrlTemplate* element or one or more *Url* elements.
- If a *UrlTemplate* element is present, each *UrlTemplate* element shall contain
    - either exactly one *id* attribute. In this case, the *SegmentInfoDefault* element on Period level shall contain a *sourceUrlTemplatePeriod* attribute with further restrictions as specified in section 12.2.4.2.2. The *duration* attribute shall be present either in the *SegmentInfo* element or in the *SegmentInfoDefault* element.
    - or exactly one *sourceURL* attribute, specifying the URL construction for Media Segments in this Representation. In this case, the *duration* attribute shall be present either in the *SegmentInfo* element or in the *SegmentInfoDefault* element.
- *Url* elements provide a set of explicit URL(s), each of which may contain a *range* attribute, for Media Segments. If more than one *Url* element is provided in the *SegmentInfo* element, the *duration* attribute shall be present either in the *SegmentInfo* element or in the *SegmentInfoDefault* element, where as in case for a single *Url* element, the *duration* attribute may not be present.
- If the duration of the last Media Segment of any Representation in a Period is significantly shorter than the value of the *duration* attribute for this Representation, or if the *duration* attribute for this Representation is not present, then MPD shall either include at least the *start* attribute of the next Period or, in case this is the last Period in the Media Presentation, include the *mediaPresentationDuration* attribute.

12.2.4.2.2 Template-based Media Segment URLs

The *SegmentInfo* Element may contain a *UrlTemplate* element and the *SegmentInfoDefault* element may contain a *sourceUrlTemplatePeriod* attribute. The *sourceURL* attribute of the *UrlTemplate* element or the *sourceUrlTemplatePeriod* attribute represents a string that contains one or more of the identifiers as listed in Table 12.1. The *sourceUrlTemplatePeriod* attribute, when present, shall contain both the *$RepresentationID$* identifier and the *$Index$* identifier. The *sourceURL* attribute, when present, shall contain the *$Index$* identifier and shall not contain the *$RepresentationID$* identifier.

A sub-string "$<Identifier>$" shall name a substitution placeholder matching a mapping key of "<Identifier>". In the request URL, the substitution placeholder shall be replaced by the substitution parameter as defined in Table 12.1. Substitution is performed left to right and identifier matching is case-sensitive. Unrecognized identifiers shall cause the URL formation to fail. In this case the client shall ignore the *Representation* element and processing of the MPD shall continue as if this *Representation* element was not present.

Table 12.1 Identifiers for URL Templates

$<Identifier>$	**Substitution parameter**
$$	Is an escape sequence, i.e. "$$" is replaced with a single "$"
$RepresentationID$	This identifier is substituted by the attribute *id* of the requested Representation in the MPD.
$Index$	This identifier is substituted by the *Index*, where the Media Segments within a Representation have assigned consecutive Segment indices from *startIndex* to *endIndex*. For an example client using this identifier to construct the list Media Segment URLs, refer to

	section 12.6.3.

12.2.5 Media Presentation Description

12.2.5.1 Introduction

The Media Presentation Description (MPD) contains metadata required by the client to construct appropriate URLs to access Segments and to provide the streaming service to the user.

The Media Presentation may be available in different Representations (different bitrates, languages, resolutions, etc.). The service may be on-demand or live. The MPD contains information that enables the client to build the URLs to access any available Segment (or parts thereof) of the Media Presentation.

The MPD is an XML-document that is formatted according to the XML schema provided in clause 12.2.5.3. The MIME type of the MPD shall be 'video/vnd.3gpp.mpd'. The delivery of the MPD is not in scope of this specification.

If the MPD is delivered over HTTP, then the MPD may be content encoded for transport, as described in [17] using the generic GZip algorithm RFC 1952 [59]. HTTP Streaming clients shall support GZip content decoding of the MPD when delivered over HTTP (GZIP RFC 1952 [59], clause 9).

An adaptive HTTP streaming client shall ignore any XML attributes or elements in a valid XML document formatted according to the XML schema provided in clause 12.2.5.3 that they do not recognize.

12.2.5.2 Media Presentation Description Attributes and Elements

The MPD contains attributes and elements as presented in Table 12.2.

Table 12.2 Semantics of Media Presentation Description (M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory)

Element or Attribute Name	Type (Attribute or Element)	Cardinality	Optionality	Description
MPD	E	1	M	The root element that carries the Media Presentation Description for a Media Presentation.
type	A		OD default: OnDemand	'OnDemand' or 'Live'. Indicates the type of the Media Presentation. Currently, on-demand and live types are defined. If not present, the type of the presentation shall be inferred as OnDemand.
availabilityStartTime	A		CM Must be present for type='Live'	Gives the availability time (in UTC format) of the start of the first Period of the Media Presentation.
availabilityEndTime	A		O	Gives the availability end time (in UTC format). After this time, the Media Presentation described in this MPD is no longer accessible. When not present, the value is unknown.
mediaPresentationDuration	A		O	Specifies the duration of the entire Media Presentation. If the attribute is not present, the duration of the Media Presentation is unknown.
minimumUpdatePeriodMPD	A		O	Provides the minimum period the MPD is updated on the server. If not present the minimum update period is unknown.
minBufferTime	A		M	Provides the minimum amount of initially buffered media that is needed to ensure

				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.

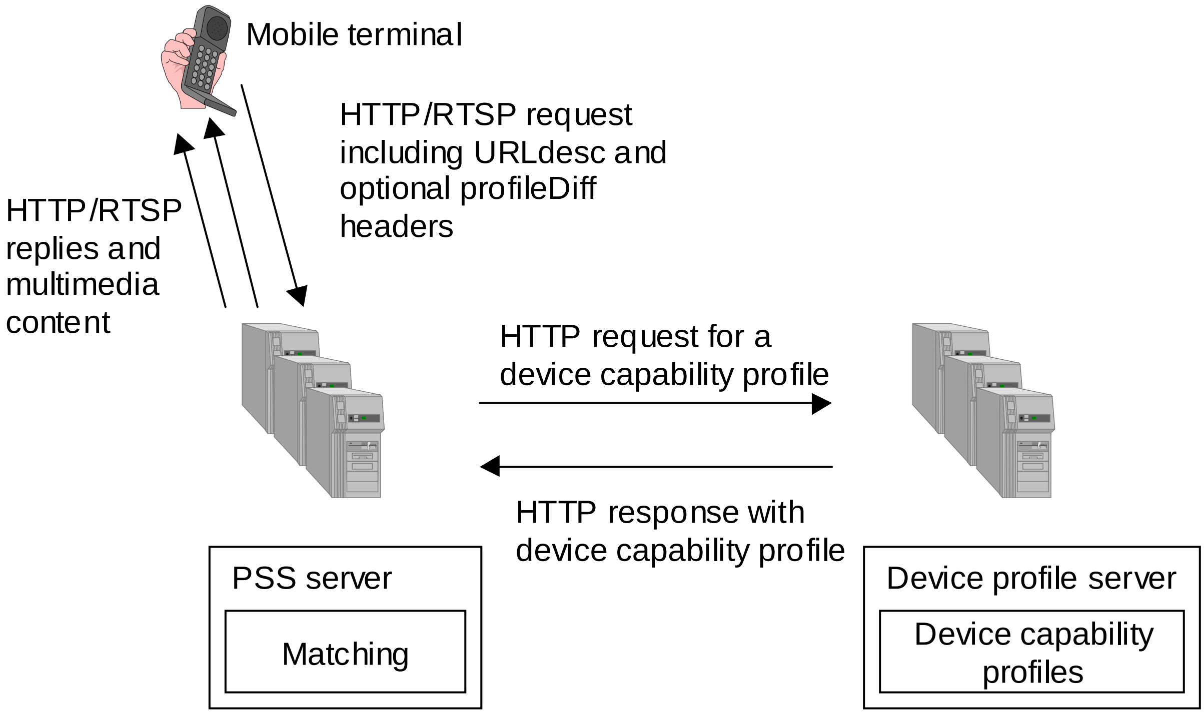


Figure A.1: Functional components in PSS capability exchange

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.

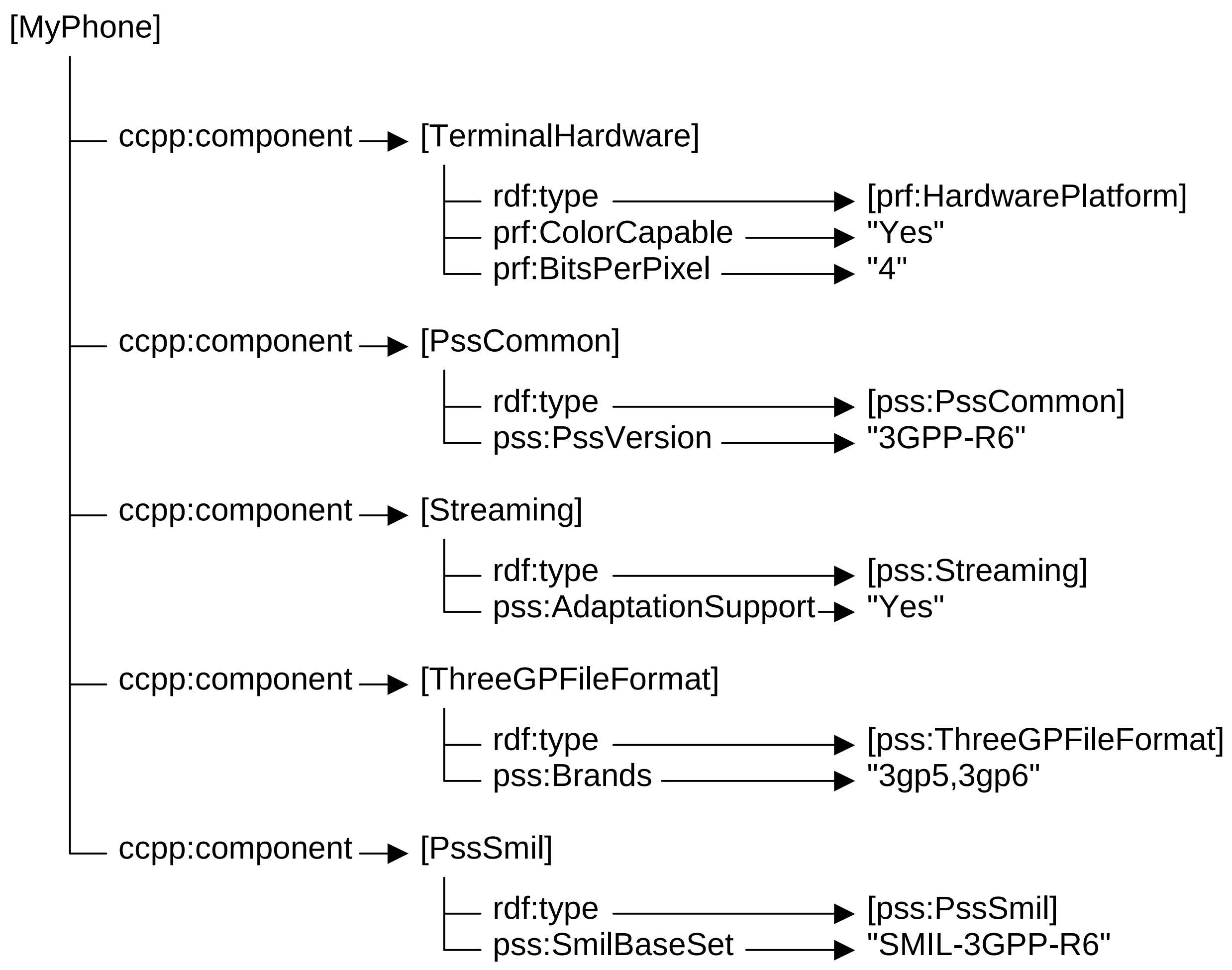


Figure A.2: Illustration of the profile structure

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.

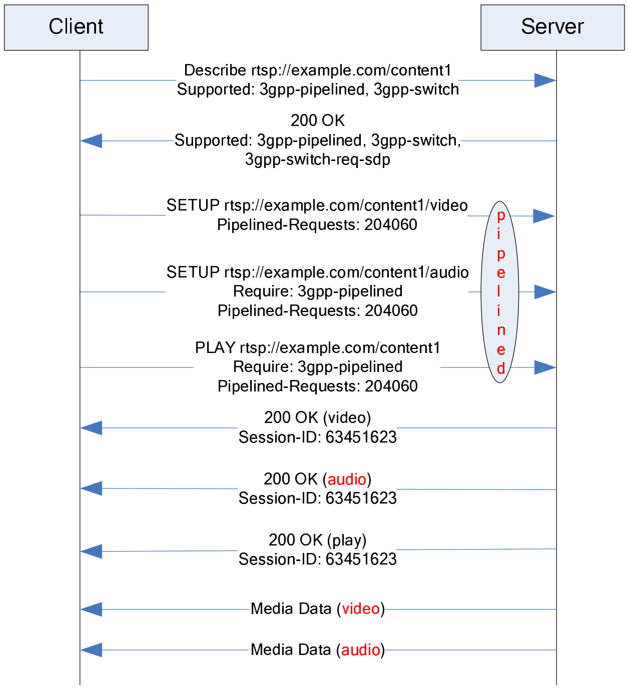
Recommendation: Authors should use property inheritance with caution.

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

M.1 Pipelined Start-up Examples

M.1.1 Successful Pipelined Start-up

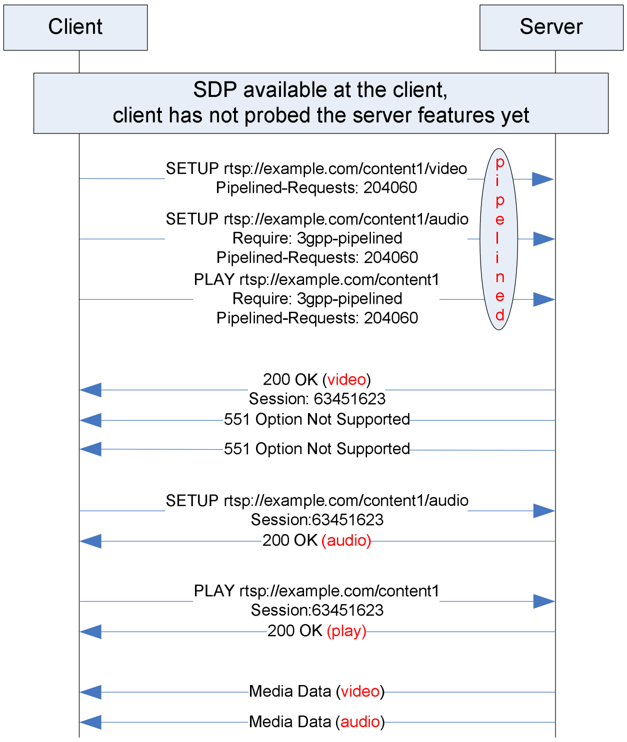
The following is an example depiction of pipelined start-up. The client probes the server features during the optional DECRIBE interaction. Note that the first set-up message does not contain a "Require" header.



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.

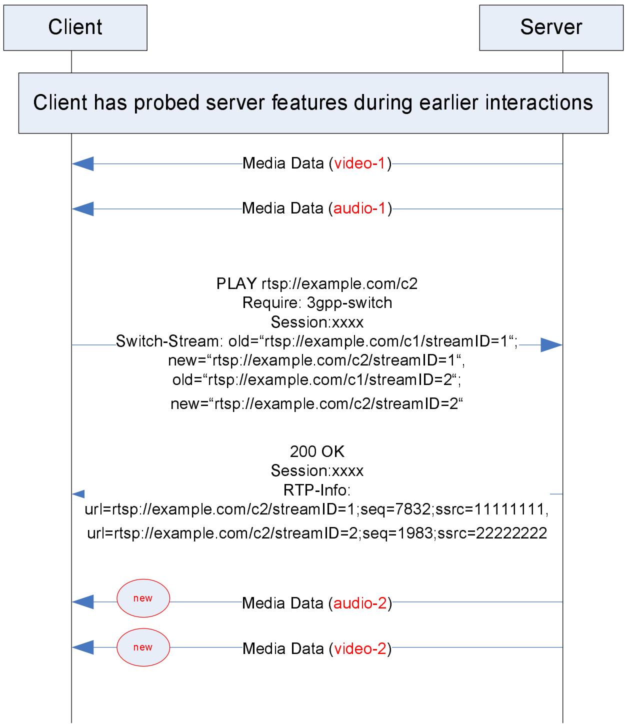
Note the first setup interaction is successful, since the client has not used the require header.



M.2 Content Switch with SDP

M.2.1 Successful Content Switch with available SDP

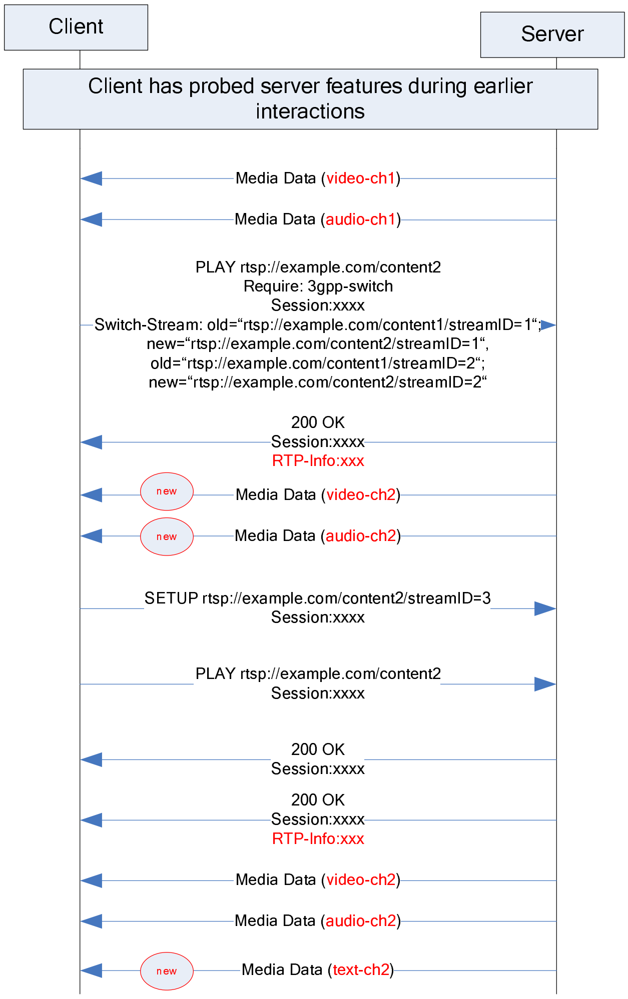
The following example depicts fast content switching with an available SDP. The client has retrieved the SDP prior to the content switch and has probed the server features during an earlier interaction.



M.2.2 Partial successful Content Switch with available SDP

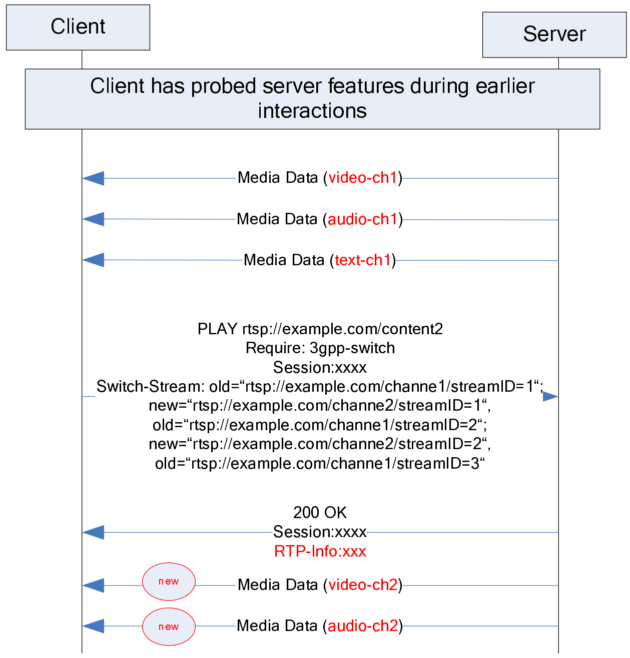
The following example depicts fast content switching with an available SDP. The content in the new SDP contains additional media components. The client has retrieved the SDP prior to the content switch and has probed the server features during an earlier interaction.

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

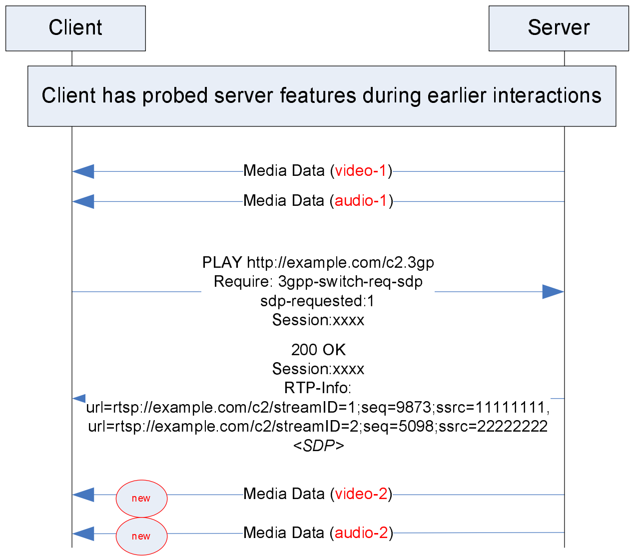
The following example depicts fast content switching with an available SDP. The content in the new SDP contains fewer media components than the previous content. The client has retrieved the SDP prior to the content switch and has probed the server features during an earlier interaction.



M.3 Content Switch without SDP

M.3.1 Successful Content Switch without available SDP

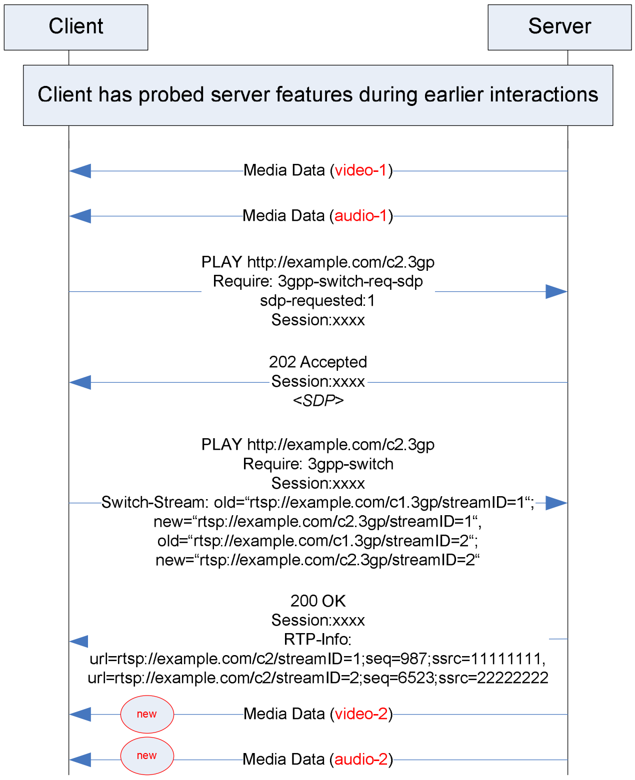
The following example depicts fast content switching without an available SDP. The client has received a content URL but has not yet retrieved the SDP. The client has probed the server features during an earlier interaction.



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.

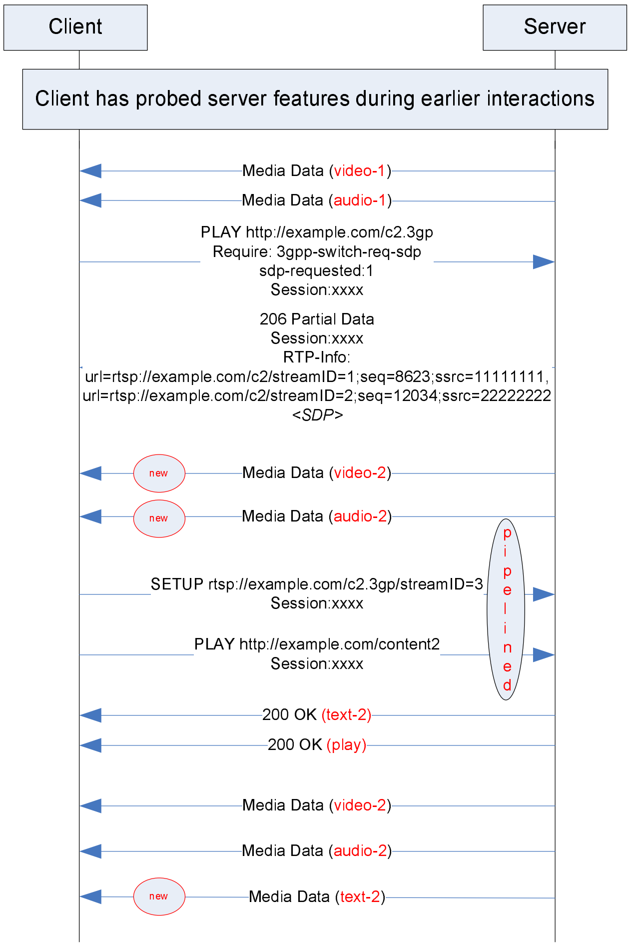
In this example, the switch request (PLAY request) returns a '202 Accepted' since the server was not able to select the desired flows from the SDP files. For example, the SDP file may contain several audio tracks and the server was unable to make a selection.



M.3.3 Partial successful Content Switch without available SDP

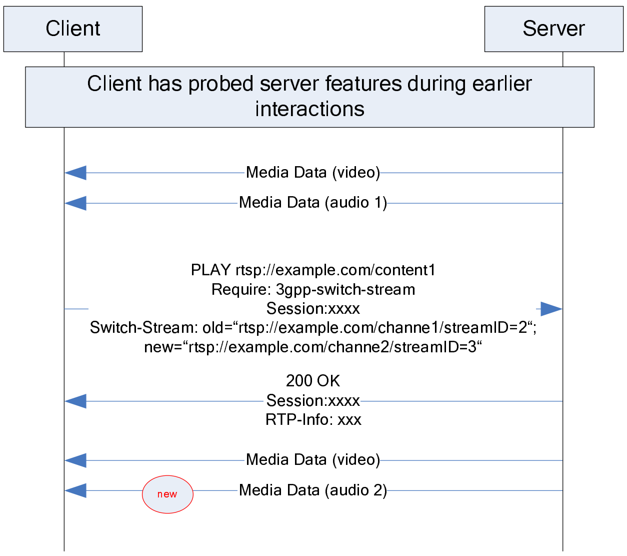
The following example depicts fast content switching without an available SDP. The client has only a content URL and still must retrieve the content description. The client has probed the server feature capabilities during an earlier interaction.

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)

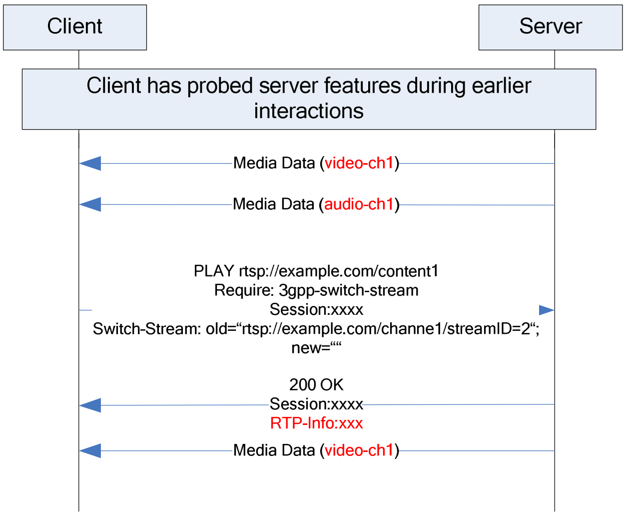


M.4 Stream Switching

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.

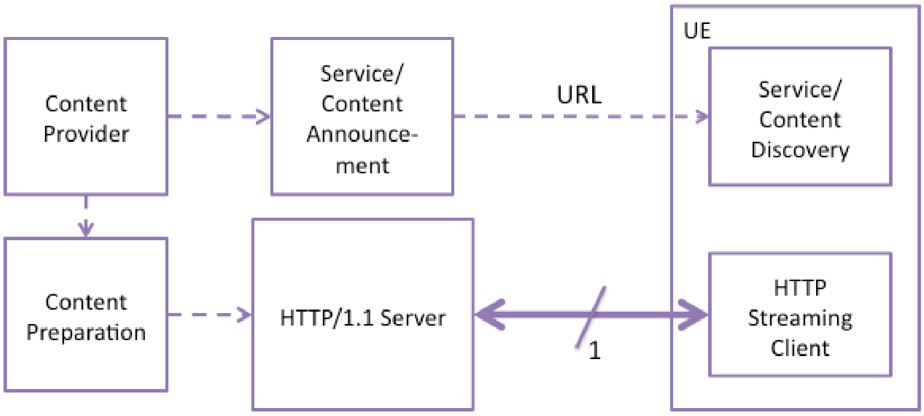


Figure Q.1 HTTP-Streaming Architecture for the Static Content Serving Mode

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.

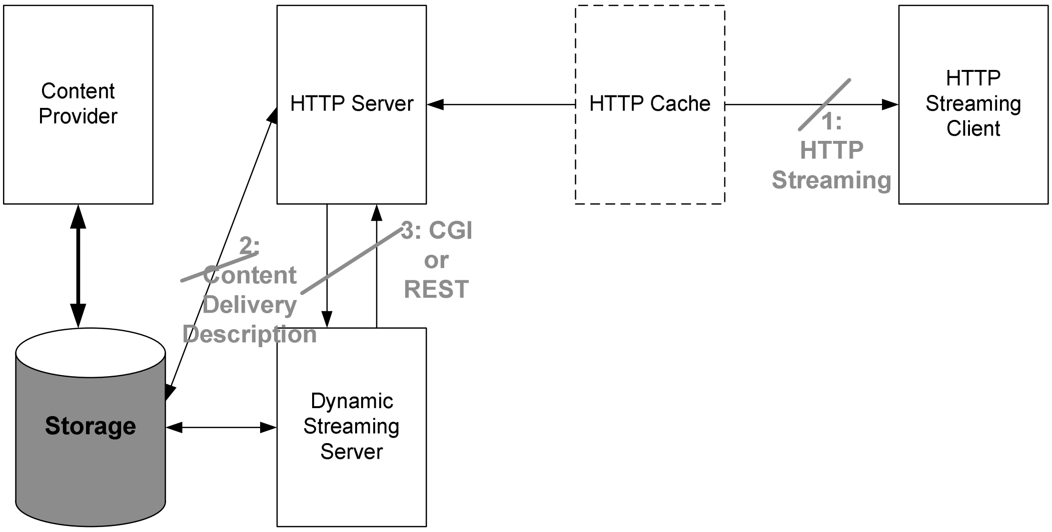


Figure Q.2 HTTP-Streaming Architecture for the Dynamic Content Serving Mode

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, ProgramInfo rmation		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